VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2020 Bc. Tomáš Dvonč

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATICKÁ DETEKCE UDÁLOSTÍ VE FOTBALOVÝCH ZÁPASECH
AN AUTOMATIC FOOTBALL MATCH EVENT DETECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Tomáš Dvonč**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Přinosil, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Bc. Tomáš Dvonč ***ID:*** 173974

Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Automatická detekce událostí ve fotbalových zápasech

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte metody vhodné pro automatickou detekci událostí z video sekvencí zaměřených na sportovní utkání (např. fotbalové zápasy). V rámci toho se seznamte s postupy a metodami pro extrakci informací z videa potřebnými pro samotnou detekci. Vybrané metody implementujte, přičemž získané informace z video sekvencí budou sloužit jako vstup pro algoritmus automatické detekce událostí. Z dostupných video sekvencí připravte trénovací a testovací množiny dat. Na základě získaných dat implementujte řešení dle návrhu tak, aby bylo schopno automaticky rozpoznat vybrané události (např. gól, start a konec zápasu či rohový kop). Vyhodnoťte úspěšnost implementovaného řešení a navrhněte jeho případná vylepšení.

DOPORUČENÁ LITERATURA:

[1] JI, Shuiwang, et al. 3D convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence, 2012, 35.1: 221-231.

[2] SHI, Yemin, et al. Sequential deep trajectory descriptor for action recognition with three-stream CNN. IEEE Transactions on Multimedia, 2017, 19.7: 1510-1520.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Jiří Přinosil, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto diplomová práca popisuje metódy vhodné na automatickú detekciu udalostí z video sekvencií zameraných na futbalové zápasy. Prvá časť práce je zameraná na analýzu a vytvorenie postupov na extrakciu informácií z dostupných dát. Druhá časť práce sa zaoberá implementáciou vybraných metód a algoritmu neurónovej siete pre detekciu rohového kopu. V práci boli realizované dva experimenty. Prvý zachytáva statickú informáciu z jednej snímky a druhý je zameraný na detekciu z časopriestorových dát. Výstupom tejto práce je program na automatickú detekciu udalostí, pomocou ktorého je možné interpretovať výsledky z vytvorených experimentov. Táto práca môže figurovať ako základ k získaniu nových poznatkov o problematike a takisto k ďalšiemu vývoju detekcie udalosti z futbalu.

KĽÚČOVÉ SLOVÁ

futbal, rozpoznávanie akcie, hlboké učenie, 2D, 3D, CNN, RNN, tensorflow

ABSTRACT

This diploma thesis describes methods suitable for automatic detection of events from video sequences focused on football matches. The first part of the work is focused on the analysis and creation of procedures for extracting informations from available data. The second part deals with the implementation of selected methods and neural network algorithm for corner kick detection. Two experiments were performed in this work. The first captures static information from one image and the second is focused on detection from spatio-temporal data. The output of this work is a program for automatic event detection, which can be used to interpret the results of the experiments. This work may figure as a basis to gain new knowledge about the issue and also to the further development of detection events from football.

KEYWORDS

football, action recognition, deep learning, 2D, 3D, CNN, RNN, tensorflow

DVONČ, Tomáš. *Automatická detekce událostí ve fotbalových zápasech*. Brno, 2020, 64 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Jiři Přinosil, Ph.D.

VYHLÁSENIE

Vyhlasujem, že svoju diplomovú prácu na tému „Automatická detekce událostí ve fotbalových zápasech“ som vypracoval samostatne pod vedením vedúceho diplomovej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Jiřimu Přinosilovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Úvod **11**

1 Detekcia udalostí vo futbale **12**
1.1 Motivácia a využitie 12
1.2 Úvod do problematiky 13

2 Hlboké učenie **14**
2.1 2D konvolučné neurónové siete 14
2.1.1 GoogLeNet Inception model 16
2.2 3D konvolučné neurónové siete 18
2.3 Rekurentné neurónové siete LSTM 20
2.4 Rozpoznávanie akcie z videa 21
2.4.1 Jednoprúdové konvolučné neurónové siete 24
2.4.2 Dvojprúdové konvolučné neurónové siete 26
2.4.3 Optický tok 27

3 Predspracovanie vstupných dát **28**
3.1 Analýza vstupných dát 29
3.2 Orezanie oblasti z video sekvencií 30
3.3 Segmentácia video sekvencií 32
3.3.1 Pohybový detektor 32
3.3.2 Generátor rohových kopov 32
3.4 Extrakcia snímok z videosekvencie 33
3.4.1 Metóda akumulácie snímok 34
3.5 Augmentácia dát 36

4 Návrh systému na detekciu udalostí **37**
4.1 Zoznámenie sa s nástrojom Docker 38
4.2 Implementácia algoritmov neurónovej siete 40
4.2.1 Inception model 41
4.2.2 3D konvolučná sieť 42
4.2.3 LSTM rekurentná sieť 43

5 Výsledky a experimenty **44**
5.1 Príprava datasetov 44
5.2 Experiment č. 1: statická detekcia obrazu 45
5.2.1 2D konvolúcia pomocou Inception modelu 45
5.3 Experiment č. 2: zachytenie pohybovej zložky 48

5.3.1 3D konvolučné neurónové siete 48
5.3.2 LSTM rekurentná neurónová sieť 50
5.4 Experiment č. 3: akumulácia snímok 51
5.5 Porovnanie experimentov 52

6 Detektor udalostí **54**
6.1 Anotácia udalostí z videozáznamov 55

Záver **56**

Literatúra **57**

Zoznam symbolov, veličín a skratiek **60**

Zoznam príloh **61**

A Návod na spustenie aplikácie **62**
A.1 Predspracovanie dát 62
A.2 Neurónová sieť 63
A.3 Detektor 63

B Obsah priloženého CD **64**

Zoznam obrázkov

1.1 Analýza defenzívneho štýlu 12
2.1 Zjednodušený model založený na CNN 14
2.2 Architektúra 2D konvolučných neurónových sietí 15
2.3 Zjednodušený Inception modul (bez 1×1 konvolúcie) 16
2.4 Ukážka operácie 5×5 konvolúcie (bez 1×1 konvolúcie) 16
2.5 Ukážka operácie 5×5 konvolúcie (s použitím 1×1 konvolúcie) . . . 17
2.6 Inception modul s použitím 1x1 konvolúcie 17
2.7 Princíp operácie 2D a 3D konvolúcie 18
2.8 Architektúra 3D konvolučných neurónových sietí 19
2.9 Bunka LSTM a jej operácie 21
2.10 Sekvencia snímok pokutového kopu 22
2.11 Detekcia objektov z vizuálnej časti videa 23
2.12 Prístupy fúzie časovej dimenzie prostredníctvom siete 25
2.13 Architektúra dvojprúdovej konvolučnej neurónovej siete 26
2.14 Vektor optického toku pohybujúceho sa objektu 27
2.15 Princíp optického toku 27
3.1 Ukážka hracej plochy zo snímky videozáznamu 29
3.2 Ukážka výrezu videozáznamu s oblasťou rohového kopu 30
3.3 Proces validácie súradníc počiatočného bodu výrezu 31
3.4 Proces vytvorenia masky pomocou odčítania pozadia 34
3.5 Výsledok akumulácie z 25 snímok video sekvencie 35
3.6 Vytvorenie novej snímky po procese augmentácie 36
4.1 Jednotlivé časti vytvoreného nástroja na detekciu udalosti 37
4.2 Realizácia kontajnerov v nástroji Docker 38
4.3 Ukážka súboru Dockerfile s tensorflow 39
5.1 Priebeh trénovania modelu Inception 46
5.2 Priebeh konvergencie validačnej chyby modelu Inception 46
5.3 Graf ROC krivky 47
5.4 Priebeh trénovania modelu 3D konvolučnej siete 49
5.5 Priebeh konvergencie chybovej funkcie modelu 3D konvolučnej siete . 49
5.6 Priebeh konvergencie chybovej funkcie modelu LSTM 50
5.7 Priebeh konvergencie chybovej funkcie pri akumulácií snímok 51
6.1 Praktická ukážka procesu posuvného okna 54
6.2 Ukážka z výstupu detektora udalostí 55

Zoznam tabuliek

3.1 Ukážka dát súboru z výstupu extrahovania snímok 33
5.1 Distribúcia dát naprieč typu udalosti 44
5.2 Hodnoty použitých parametrov na trénovanie modelu Inception . . . 45
5.3 Hodnoty použitých parametrov na trénovanie 3D konvolučnej siete . . 48
5.4 Hodnoty použitých parametrov na trénovanie LSTM siete 50
5.5 Hodnoty použitých parametrov na trénovanie LSTM siete 51
5.6 Dostupný hardware na trénovanie neurónovej siete 52
5.7 Porovnanie výsledkov z oboch experimentov 53

Zoznam výpisov

3.1 Proces segmentácie videoklipov 32
3.2 Horizontálne preklopenie videoklipu 33
3.3 Implementácia vytvorenia akumulovanej snímky 35
4.1 Implementácia modelu neurónovej siete Inception 41
4.2 Implementácia 3D konvolučnej neurónovej siete 42
4.3 Extrakcia príznakov pomocou Inception modelu 43
4.4 Implementácia LSTM modelu neurónovej siete 43
5.1 Prikaz vytvorenia trénovacej a testovacej dátovej sady 44

Úvod

Svet športu predstavuje jeden z najsledovanejších obsahov v digitálnej televízii a na webe. Športové udalosti sledujú každodenne milióny ľudí a prevádzkovatelia vysielania sa snažia každodenne zlepšovať poskytovanie štatistík hry v reálnom čase.

Vývoj automatických metód analýzy akcií vo videách má osobitný význam v komerčnej aj v profesionálnej sfére. Športoví analytici, komentátori môžu v reálnom čase analyzovať a komentovať priebeh športovej udalosti. Tréneri môžu ľahko analyzovať individuálnu hru každého hráča a zároveň pripraviť taktiku proti svojmu súperovi. Ďalším využitím je zozbieranie štatistických údajov o hre, ako napríklad počet striel na bránu alebo rohové kopy. Na konci sezóny môžu byť tieto údaje použité na vytvoreniu štatistiky jednotlivých tímov alebo hráčov.

Naneštastie, pri samotnej realizácii detekcie udalosti do hry vstupujú aj rôzne nepriaznivé aspekty, ktoré môžu svojím pôsobením narušiť efektivitu zvolenej metodiky. Medzi tieto aspekty je možné zaradiť prírodné podmienky alebo komplexitu a variabilitu herného štýlu jednotlivých hráčov. Niekedy vzniknú situácie, keď kamera nie je schopná zaznamenať všetky potrebné objekty. Napríklad lopta môže byť zakrytá telom hráča a podobne.

Táto práca sa zaoberá detekciou udalostí z futbalových zápasov. Najdôležitejšou časťou práce bolo získať, analyzovať a spracovať dostupné sekvencie videozáznamov na vstup do neurónovej siete. Následne sa tieto znalosti použili k vytvoreniu nástroja umelej inteligencie, so schopnosťou správne klasifikovať udalosti do jednotlivých tried.

Hlavným prínosom prace je poskytnúť prehľad o problematike detekcie udalostí zo športových zápasov a na základe poznatkov navrhnúť optimálne riešenie. Samotná práca môže byť v prvotnej fáze vývoja použitá ako nástroj na vytvorenie anotácií z videozáznamu.

V prvej kapitole je krátko vysvetlený úvod do problematiky, zatiaľ čo druhá kapitola poskytuje teoretické rozšírenie poznatkov v oblasti rozpoznávania akcie z videa. Detailnejšie sú rozobraté jednotlivé typy neurónových sietí vhodných na úlohu rozpoznávania udalostí.

Ďalšie kapitoly sú venované praktickej realizácii práce, čo zahŕňa predspracovanie vstupných dát a návrh algoritmu neurónovej siete. Následne budú zverejnené výsledky experimentov.

Posledná kapitola v krátkosti popisuje nástroj vytvorený na anotáciu jednotlivých udalostí z videozáznamov.

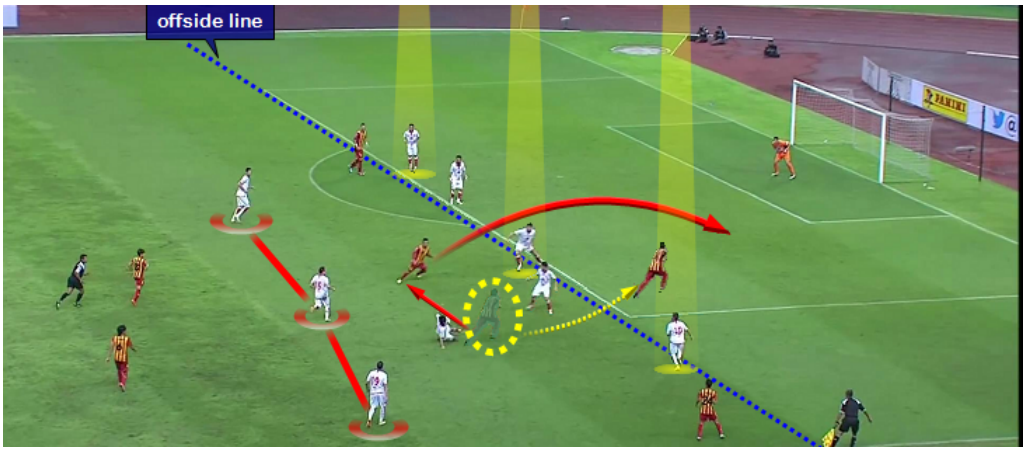
1 Detekcia udalostí vo futbale

Začiatkom 20. storočia sa šport zmenil z obľúbenej a nevinnej hry na veľký biznis. V spoločnosti ma čoraz väčší význam vzhľadom na technologický pokrok, ktorý umožňuje sprístupňovať živé vysielanie každému.

Táto kapitola je zameraná na objasnenie problematiky detekcie udalostí vo futbale, pričom bude vysvetlená motivácia a využitie takéhoto systému.

1.1 Motivácia a využitie

Keďže v športe je aktuálne zahrnuté veľké množstvo financií, kluby sa snažia urobiť maximum pre to, aby sa dostali na vrchol. Na dosiahnutie lepšieho výkonu je potrebné monitorovať rôzne aspekty hry – **sofistikovaná športová analytika**.



Obr. 1.1: Taktická analýza defenzívneho herného štýlu [1]

Pohyb jednotlivých hráčov v oblasti športu je komplexný a ovplyvňujú ho rôzne faktory, ako napríklad taktika alebo kontext hry. Hráči sa musia formou tréningu pripravovať na konkrétne zápasy a situácie. Tréneri často prispôsobujú herný štýl v závislosti od toho, aký súper ich čaká nasledujúci zápas. Preto rozpoznanie tímovej hry súperovho tímu je veľmi užitočné. Hráči spolu s trénerom môžu analyzovať hru a slabiny súperovho tímu. Tieto informácie môžu byť dôležité pre vytvorenie taktiky na nasledujúci zápas.

V súčasnosti profesionálni tréneri získavajú túto identifikáciu ručným priraďovaním anotácií ku konkrétnym udalostiam ako napríklad strely, góly, rohové kopy a podobne. Ako príklad tejto analýzy môže slúžiť agregácia počtu gólov, faulov, prípadne počtu striel. Tréner na základe tejto informácie môže s určitou presnosťou definovať herný štýl tímu a pripraviť na to svoj tím. Získavanie týchto informácií sa

často realizuje prostredníctvom ľudských odborníkov. Tí manuálne zanalyzujú celý videozáznam a pripravia pre trénera štatistiky. Tento postup je však časovo náročný, zdĺhavý a môže spôsobiť chyby v dôsledku subjektívnej povahy manuálnej anotácie.

Riešením tohto problému a odľahčením práce pre ľudských pracovníkov je použitie nástroja umelej inteligencie, ktorý je rýchlejší a má výrazne nižšie finančné náklady na správu. Za pomoci umelej inteligencie je možné vytvoriť plne automatizovaný nástroj, ktorý by z obrazu rozpoznával dôležité udalosti. Výstupom tohto systému by mohla byť štatistika týchto udalostí – **počet gólov**, **faulov**, **rohových kopov** atď.

V minulosti vznikli rôzne projekty s týmto zámerom, kde sa podarilo docieliť metodiku schopnú vykonávať úlohu detekcie aktivity tímu s vysokou presnosťou pri klasifikácii troch futbalových akcií: držanie lopty, protiútok, formácia hráčov [2].

1.2 Úvod do problematiky

Cieľom diplomovej práce je pripraviť automatizovaný nástroj s použitím umelej inteligencie, ktorý je schopný detektovať rohové kopy z futbalových zápasov. Veľmi dôležitú časť tejto práce tvorí vhodné predspracovanie vstupných dát. Zlá manipulácia s dátami môže ovplyvniť prácu s algoritmom pre automatickú detekciu udalostí, a preto je dôležité venovať tejto časti dostatočné úsilie.

Prvým dôležitým aspektom vytvorenia komplexnej neurónovej siete je vytvorenie **variabilných** vstupných dát. Pri nedodržaní tohto princípu by sa mohlo stať, že sieť bude síce s veľkou presnosťou rozpoznávať danú variáciu vstupných dát, ale pri rôznorodosti testovacej množiny nebude schopná správne klasifikovať tieto prvky. Preto variabilita vstupných dát je veľmi dôležitá pri regularizácii neurónovej siete.

Ďalší faktor, ktorý môže ovplyvniť chod siete, je množstvo vstupných dát. Platí pravidlo – čím viac dát, tým lepšie sa sieť dokáže adaptovať na nové dáta.

Pri realizácii neurónovej siete treba takisto zohľadniť niekoľko faktorov, ktoré môžu byť v určitých situáciách nežiaduce. V kontexte tejto práce bude natrénovaná neurónová sieť z množiny video sekvencii futbalových zápasov, z ktorých sa extrahujú krátke segmenty videa. Pre správne natrénovanie siete je dôležité, aby dáta boli rovnomerne rozdelené medzi všetky triedy, ktoré budú podliehať klasifikácii. Napríklad, ak by jedna z tried obsahovala len zlomok dát v porovnaní s ostatnými, tak vo výsledku to môže značne ovplyvniť správanie siete v tejto triede.

2 Hlboké učenie

Pojem hlboké učenie sa dá špecifikovať ako technika strojového učenia, ktorá učí počítač robiť to, čo je pre človeka prirodzené – učiť sa príkladom. Pri hlbokom učení sa počítačový model učí vykonávať klasifikačné úlohy priamo z obrázkov, videa, textu alebo zvuku. Výsledky hlbokého učenia môžu niekedy dosahovať presnosť presahujúcu úroveň človeka.

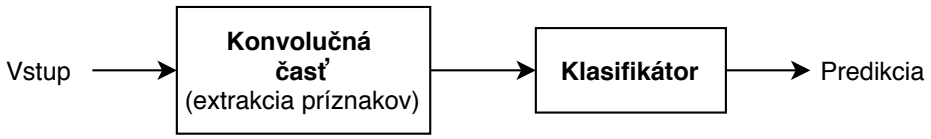
Táto kapitola detailnejšie popisuje teóriu problematiky rozpoznávania akcie z videa. Na začiatku budú charakterizované tri rôzne metódy prístupu detekcie udalostí, ktoré boli implementované v praktickej časti diplomovej práce. Nakoniec budú vysvetlené dva spôsoby realizácie neurónových sietí vzhľadom na definíciu pohybovej a časovej zložky.

2.1 2D konvolučné neurónové siete

Prvým možným prístupom pri realizácií detekcie udalosti z videa je použitie 2D konvolučných neurónových sietí. Základným princípom tejto metódy je statická analýza objektov a pozadia v určitej snímke. Vo výsledku ide o to, aby bol v každej snímke zachytený nejaký špecifický kontext, ktorý je charakteristický pre danú udalosť. Ako príklad je možné uviesť hru na gitare, kde sémantický obsah udalosti tvorí osoba a konkrétny hudobný nástroj.

Konvolučné neuronové siete (*ConvNets*) sú špeciálne typy neurónových sietí, ktoré sú prispôsobené aplikáciám počítačového videnia tak, aby boli schopné abstrahovať lokálne reprezentácie obrazu. Je to špecializovaný typ modelu neurónovej siete, ktorý je navrhnutý na prácu s dvojrozmernými obrazovými údajmi, hoci sa môžu použiť s jednorozmernými a trojrozmernými dátami. Architektúra konvolučných neurónových sietí sa skladá z troch hlavných blokov:

1. **Konvolučná vrstva**, ktorá extrahuje funkcie z zdrojovej snímky
2. **Pooling vrstva**, ktorá podvzorkuje každú funkciu, aby sa zmenšil jej rozmer a zachytila sa tá najdôležitejšia zložka.
3. **Plne prepojená vrstva**, ktorá spája prvky identifikované v predchádzajúcich vrstvách na vektor, pričom predikuje kategóriu výstupu

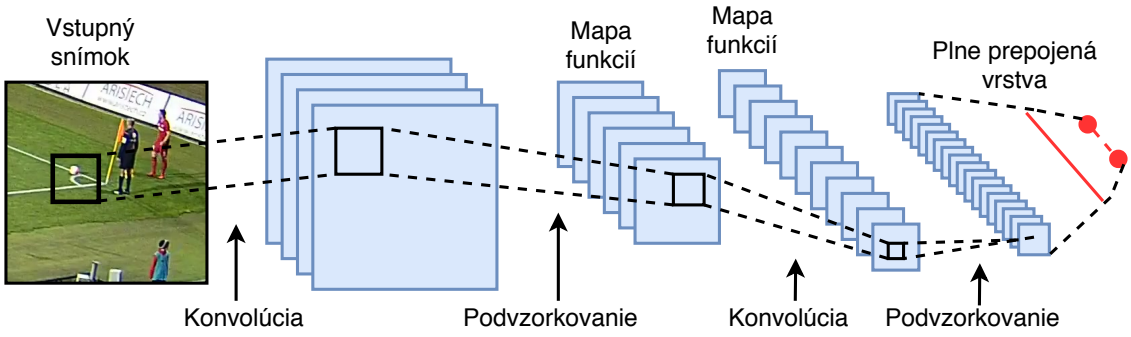


Obr. 2.1: Zjednodušený model založený na CNN

Konvolučné vrstvy sú hlavným stavebným blokom konvolučných neurónových sietí. Stručne povedané, konvolučná vrstva skenuje zdrojový obrázok s filtrom o definovanom rozmere (napr. 5×5 pixelov), aby extrahovala prvky, ktoré môžu byť dôležité na klasifikáciu. Tento filter sa nazýva **konvolučné jadro** (*Convolution Kernel*). Jadro obsahuje váhy, ktoré sú optimalizované tak, aby dosiahli čo najpresnejšie predikcie.

V konvolučných neurónových sieťach každá vrstva pôsobí ako detekčný filter na zistenie prítomnosti špecifických znakov alebo vzorov v originálnej snímke. Prvé vrstvy detekujú „veľké“ znaky, ktoré je možné pomerne ľahko rozoznať a interpretovať. Ostatné vrstvy zisťujú postupne „menšie“ črty, ktoré sú čo raz viac abstraktnejšie. Posledná vrstva umožňuje následne urobiť klasifikáciu kombináciou všetkých vlastností zistených predchádzajúcimi vrstvami vo vstupných údajoch.

Veľmi dôležitou vlastnosťou týchto filtrov je definícia konkrétneho filtra na systematickú detekciu špecifických znakov. Potom použitie tohto filtru v celom vstupnom snímku umožňuje objaviť charakteristické črty kdekoľvek v snímku. Táto vlastnosť sa bežne označuje ako **prekladová invariancia** (Translation Invariance).



Obr. 2.2: Architektúra 2D konvolučných neurónových sietí

Jedným z dôležitých aspektov týchto sietí je to, že sa dokážu automaticky učiť hierarchické reprezentácie prvkov. To znamená , že príznaky vypočítané prvou vrstvou sú všeobecné, zatiaľ čo funkcie vypočítané poslednou vrstvou sú špecifické a závislé od vstupného súboru údajov.

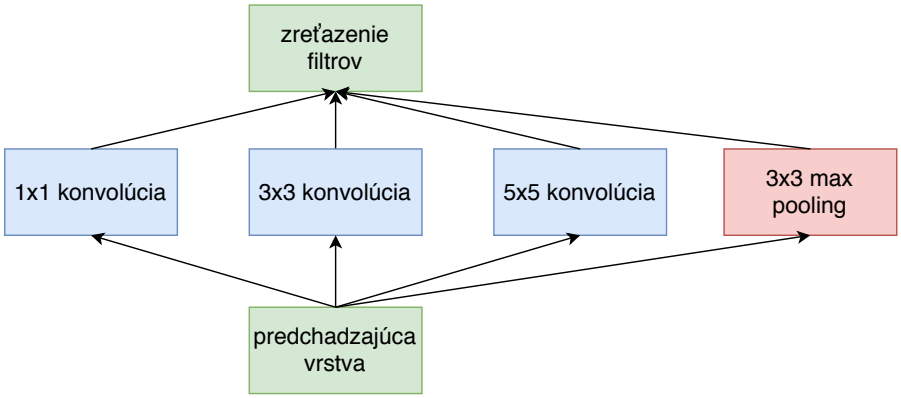
V prípade nedostatočných pamäťových prostriedkov je možné 2D konvolučnú sieť použiť aj na **extrakciu príznakov**. To je možné docieliť skompilovaním modelu s použitím vrstvy spriemerovania (*Average Pooling*). Efektívne sa oddelí klasifikačná časť siete a výstupom bude vektor s 2048 prvkami. Výstup z takto modifikovanej architektúry sa dá použiť ako vstup do samostatnej siete RNN (*Recurrent neural network*) [3].

2.1.1 GoogLeNet Inception model

Inception je široko používaný model rozpoznávania obrázkov s vopred natrénovanou hlbokou neurónovou sieťou. Túto sieť navrhol *Yann LeCun* a jeho kolegovia, pričom bola pôvodne využívaná v „LeNet“ – čo bola úspešná konvolučná neurónová sieť na identifikáciu objektov z obrázkov. Existujúcu Inception sieť je možné preškoliť, aby bola schopná použiť riešenie na úlohu klasifikácie vlastných obrázkov.

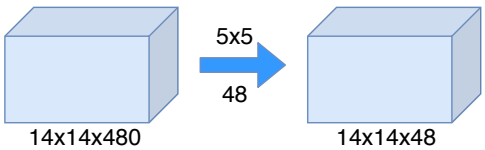
Počiatočná architektúra bola navrhnutá tak, aby fungovala dobre aj pri rôznych obmedzeniach týkajúcich sa pamäte a výpočtového výkonu.

Na obrázku 2.3 je zobrazený základný modul architektúry Inception siete. Vo výsledku sa jedna o 1×1, 3×3, 5×5 konvolúciu a *Pooling* vrstvu. Inception sieť je zaujímavá tým, že umožňuje použitie variabilnej veľkosti filtrov a zároveň použitie všetkých konvolučných vrstiev a *Pooling* vrstvy naraz, pričom sa výstup z každej vrstvy v poslednej časti zlúči do jedného celku zreťazených filtrov.



Obr. 2.3: Zjednodušený Inception modul (bez 1×1 konvolúcie)

V súčastnosti sa tento zjednodušený model (bez 1×1 konvolúcie) nepoužíva, a to z dôvodu veľkého počtu operácií, ktoré by sieť musela vykonať. Obrázok 2.4 znázorňuje použitie 5×5 konvolúcie – bez použitia prídavnej 1×1 konvolúcie. Výstupom konvolúcie je $(14 \cdot 14 \cdot 480) \cdot (14 \cdot 14 \cdot 48) =$ **112,9 miliónov** operácií, čo predstavuje obrovskú časovú a výpočtovú náročnosť.

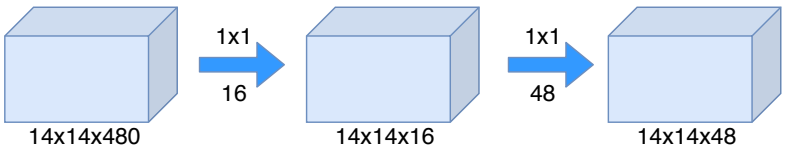


Obr. 2.4: Ukážka operácie 5×5 konvolúcie (bez 1×1 konvolúcie)

Riešením problému 5×5 konvolúcie je redukcia veľkého počtu operácií pridaním 1×1 konvolučného filtra (obr. 2.5). Tento filter pomáha k obrovskému zníženiu výpočtových požiadaviek (vysvetlené ďalej). Okrem toho samotná 1×1 konvolúcia pomáha znížiť problém s pretrénovaním siete (*overfitting*). Výsledný počet operácií tohto typu konvolúcie bude nasledovný:

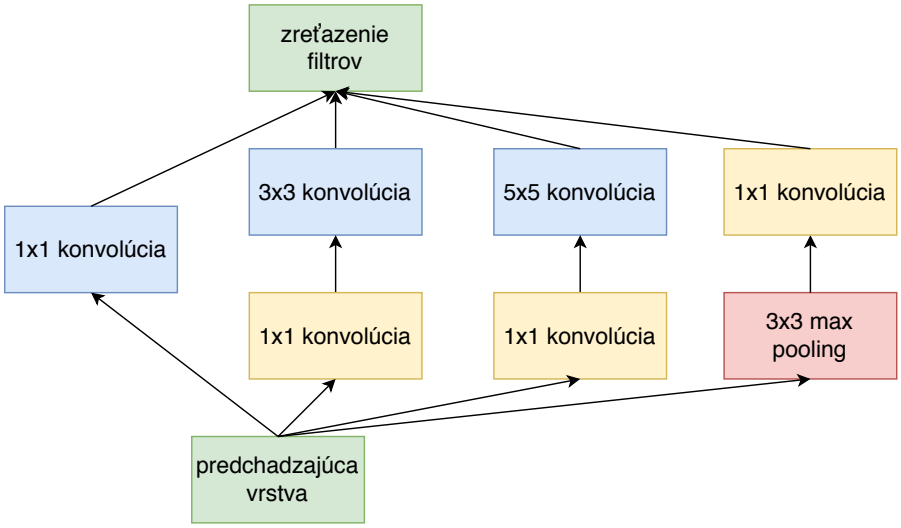
- Počet operácií po 1×1 konvolúcií: $(14 \cdot 14 \cdot 16) \cdot (1 \cdot 1 \cdot 480) =$ **1,5 milióna**
- Počet operácií po 5×5 konvolúcií: $(14 \cdot 14 \cdot 48) \cdot (5 \cdot 5 \cdot 16) =$ **3,8 milióna**

Vo výsledku je celkový počet operácií: $1{,}5\,\mathrm{M} + 3{,}8\,\mathrm{M} =$ **5,3 miliónov** operácií.



Obr. 2.5: Ukážka operácie 5×5 konvolúcie (s použitím 1×1 konvolúcie)

GoogLeNet postavil túto architektúru na myšlienke, že väčšina aktivácií neurónov v hlbokej sieti je nepotrebná (nulová hodnota) alebo redundantná kvôli vzájomným koreláciám. Navrhnutý Inception modul má teda riedke spojenie (*sparse connection*) medzi neurónmi, hlavne v konvolučných vrstvách, čo znamená, že niektoré z výstupných kanálov nebudú spojené so vstupnými kanálmi [4, 5].



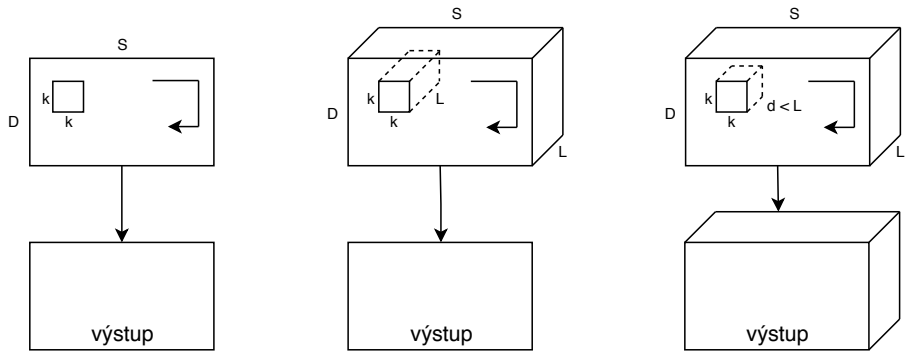
Obr. 2.6: Inception modul s použitím 1x1 konvolúcie

2.2 3D konvolučné neurónové siete

V dnešnej dobe, kedy internet rastie extrémnou rýchlosťou, je čoraz viac potrebné analyzovať a porozumieť obsahu nahrávaných dát. Celé desaťročia sa riešia rôzne problémy, ako je napríklad rozpoznávanie, detekcia abnormálnych udalostí a ich porozumenie. Značný pokrok bol dosiahnutý v týchto jednotlivých problémoch pomocou rôznych špecifických riešení. Stále však existuje potreba všeobecného **deskriptora videa**, ktorý pomáha pri riešení úloh detekcie „homogénnym“ spôsobom. Účinný deskriptor videa by mal zahŕňať štyri základné vlastnosti:

1. **Všeobecnosť** – aby dokázal dobre reprezentovať rôzne typy videí a zároveň by mal byť diskriminačný. Napríklad môžu existovať videá obsahujúce krajiny, prírodné scény, šport, televízne programy, filmy, domáce zvieratá, jedlo atď.
2. **Kompaktnosť** – keďže v súčastnosti sa pracuje s miliónmi videí, kompaktný deskriptor by mal byť ľahko škálovateľný pri spracovaní, ukladaní a získavaní nových úloh.
3. **Účinnosť** – deskriptor musí byť schopný spracovať každú minútu tisícky videí v rôznych reálnych systémoch.
4. **Jednoduchosť** – mal by byť jednoduchý na implementáciu. Namiesto používania komplikovaných metód a klasifikátorov je vyžadované, aby deskriptor vedel dobre fungovať aj s jednoduchým modelom (napr. lineárny klasifikátor)

3D konvolučná sieť je vhodná na trénovanie časopriestorových funkcií. V porovnaní s 2D konvolučnou sieťou má schopnosť lepšie modelovať časové informácie. Na obrázku 2.7 je znázornený rozdiel v procese 2D a 3D konvolúcie, pričom $\boldsymbol{D}$ a $\boldsymbol{S}$ definujú dĺžku a šírku daného snímku, $\boldsymbol{k}$ predstavuje dĺžku filtra, $\boldsymbol{L}$ špecifikuje počet snímok a $\boldsymbol{d}$ je počet filtrov. Ako je možné pozorovať, len 3D konvolúcia si uchováva časovú informáciu.

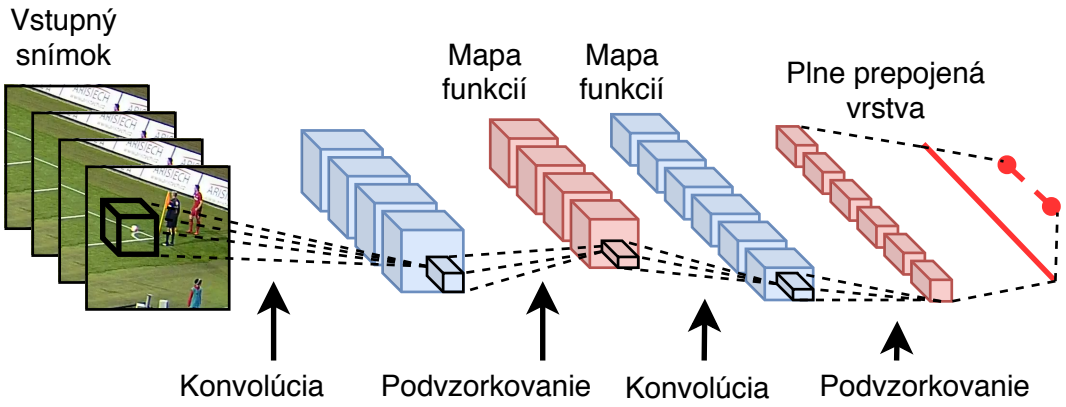


a) 2D konvolúcia b) 2D konvolúcia viacerých snímok c) 3D konvolúcia

Obr. 2.7: Princíp operácie 2D a 3D konvolúcie

Konvolučné neurónové siete sa v súčasnosti značne využívajú v oblasti počítačového videnia. CNN možno kategorizovať na základe dimenzie použitého konvolučného jadra na:

1. **2D konvolučné neurónové siete** – ktoré používajú 2D konvolučné jadrá na predpovedanie mapy segmentácie pre jednu snímku. Mapy segmentácie sú predpovedané vždy po jednom segmente a z toho dôvodu nedokážu využiť kontext zo susedných snímkov.
2. **3D konvolučné neurónové siete** – ktoré riešia tento problém pomocou 3D konvolučných jadier na vytváranie predikcií z viacerých snímok. Táto schopnosť môže viesť k zlepšeniu efektívnosti siete, ale je spojená s nárastom výpočtových nákladov v dôsledku zvýšeného počtu používaných parametrov na trénovanie tohto typu siete.



Obr. 2.8: Architektúra 3D konvolučných neurónových sietí

V 2D konvolučných neurónových sieťach sa do máp funkcií vypočítajú prvky iba z priestorových dimenzií. V prípade, že je žiaduce aplikovať analýzu videa na pohybové informácie kódované vo viacerých susedných snímkach, je potrebné zvážiť možnosť 3D konvolučných neurónových sietí. 3D konvolúcia zabezpečí výpočet prvkov z priestorových aj časových rozmerov. Na dosiahnutie 3D konvolúcie je potrebné vytvoriť 3D konvolučné jadro naskladaním viacerých susedných snímok dohromady. Takouto konštrukciou sú vytvorené mapy funkcií v konvolučnej vrstve, spojené s viacerými susednými rámcami v predchádzajúcej vrstve, čím sa zachytávajú informácie o pohybe.

Na základe vyššie popísanej 3D konvolúcie je možné vytvoriť rôzne typy architektúr 3D konvolučných neurónových sietí. V praktickej časti boli realizované konkrétne dva typy [6, 7, 8].

2.3 Rekurentné neurónové siete LSTM

Jednou z výziev rekurentných neurónových sietí (Recurrent neural network), ďalej len RNN, je spojenie predchádzajúcej informácie s aktuálnou. Napríklad predchádzajúcu snímku by bolo možné použiť na predikciu novej snímky. Sú však prípady, keď je k správnej predikcií potrebné zachytiť oveľa dlhší kontext s informáciami.

RNN má však tendenciu strácať svoju účinnosť úmerne zvyšovaniu rozdielu medzi analyzovanými údajmi a predchádzajúcimi vstupmi. To znamená, že hoci RNN je účinná pri práci so sekvenčnými vstupmi, opakujúce sa siete majú iba krátkodobú pamäť a nedokážu sa naučiť spájať tieto informácie (***Long-Term Dependencies Problem***). Tento problém rieši práve rekurentná sieť typu LSTM.

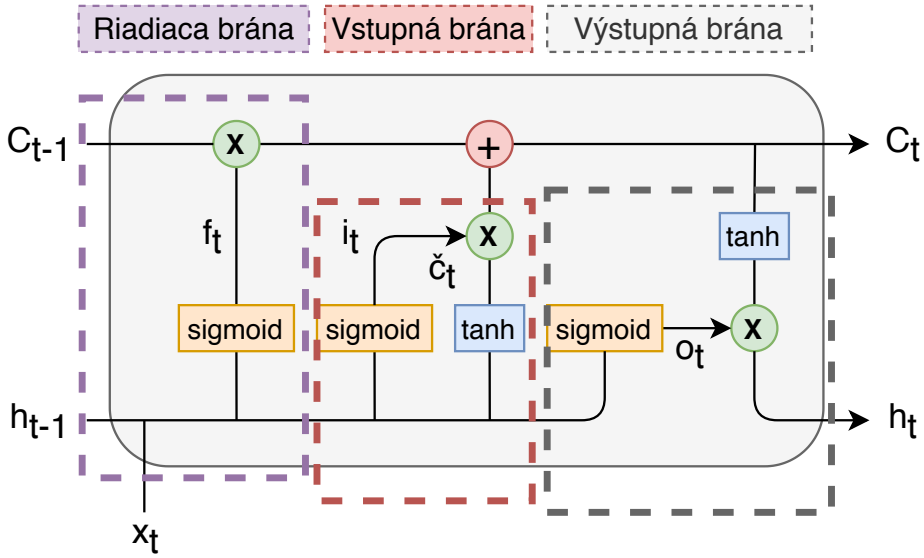
Neurónové siete typu LSTM (Long Short Term Memory) sú špeciálny druh rekurentných neurónových sietí RNN. Prvýkrát boli predstavené prostredníctvom práce od *Hochreiter* a *Schmidhuber* v roku 1997 [9]. V súčasnej dobe sa využívajú hlavne v oblasti rozpoznávania reči, generovania textu alebo titulkov pre videá.

Základným konceptom LSTM sietí je **stav buniek** (*cell state*) a vnútorné mechanizmy, nazývané **brány** (*gates*). Stav buniek pôsobí ako transportná jednotka, ktorá prenáša relatívne informácie až po celý reťazec sekvencií, tzv. „pamäť“ siete. Pri procese tréningu sa brány naučia, ktoré údaje sú relevantné a na ktoré môžu zabudnúť. Na základe toho brány rozhodnú, či daná informácia v bunke zotrvá alebo bude vymazaná.

Ako je možné vidieť na obrázku 2.9, bunka LSTM obsahuje 3 rôzne brány, ktoré majú špecifickú úlohu v procese trénovania neurónovej siete. Každá brána sa skladá zo sigmoid vrstvy neurónovej siete a operácie bodového násobenia.

1. **Riadiaca brána** (*Forget gate*) – na začiatku bunky sa nachádza riadiaca brána, ktorá rozhodne, aké informácie by sa mali zahodiť alebo ponechať. Informácie z predchádzajúceho skrytého stavu a informácie z aktuálneho vstupu sa prenášajú pomocou funkcie sigmoid. Hodnota v blízkosti 0 znamená zabudnúť a v blízkosti 1 ponechať.
2. **Vstupná brána** (*Input gate*) – aktualizáciu stavu bunky sprostredkúva vstupná brána. Na začiatku sa do sigmoid funkcie použije predchádzajúci skrytý stav a aktuálny vstup, pričom sa rozhodne o tom, ktoré hodnoty sa aktualizujú. To sa realizuje transformáciou hodnôt medzi 0 a 1, pričom 0 predstavuje menej významný stav a 1 znamená dôležitý. Skrytý stav a aktuálny vstup sa takisto prenesú do funkcie *tanh*, aby sa prepočítali hodnoty medzi -1 a 1. Nakoniec sa vynásobí výstup z *tanh* s výstupom funkcie sigmoid. Sigmoidový výstup rozhodne, ktoré informácie sú dôležité z výstupu *tanh*.

3. **Výstupna brána** (*Output gate*) – posledná brána, ktorá rozhodne, aký bude ďalší skrytý stav, ktorý obsahuje informácie o predchádzajúcich vstupoch. Na začiatku sa prevedie predchádzajúci skrytý stav a aktuálny vstup do sigmoid funkcie. Potom sa nový, modifikovaný bunkový stav posunie do funkcie *tanh*. Vynásobením *tanh* výstupu s sigmoid výstupom sa určí, aké informácie má skrytý stav obsahovať. Nový a nový skrytý stav bunky sa potom prenesú do nasledujúcej iterácie [10, 11].



Obr. 2.9: Bunka LSTM a jej operácie [11]

Legenda vysvetľujúca jednotlivé symboly definované na obrázku:

- X_t – aktuálny vstup
- C_{t-1} – aktuálny stav bunky
- h_{t-1} – skrytý stav bunky
- c_t – nový aktualizovaný stav
- h_t – aktuálny výstup
- $\times$ – operácia násobenia
- $+$ – operácia sčítania

Za pozornosť stojí výskum od *Kalchbrennera*, v ktorom popisuje implementáciu LSTM do viacrozmernej mriežky [12]. Sieť sa líši od klasických LSTM architektúr v tom, že bunky sú prepojené medzi sieťovými vrstvami, ako aj pozdĺž priestorovo-časových rozmerov údajov.

2.4 Rozpoznávanie akcie z videa

Obrázky a videá sú v dnešnej dobe veľkou súčasťou internetu. Užívatelia každú minútu nahrajú na YouTube viac ako **300 hodín** videa. Na Instagram je denne pridaných viac ako **100 miliónov** obrázkov a videí. Pre zamestnancov je nemožné v reálnom čase validovať sémantický obsah každého obrázka a videa. Vývojári preto museli aplikovať do tohto systému počítačové videnie – to naštartovalo vývoj programov na prácu s multimediálnym obsahom.

Cieľom rozpoznávania aktivity je rozpoznávanie činností jedného alebo viacerých činiteľov zo série pozorovaní v rôznych environmentálnych podmienkach. Rozpoznávanie akcie z videa vyžaduje zachytenie kontextu z celého úseku videa, nie iba zachytenie informácie z určitých snímok.

Obr. 2.10: Sekvencia následujúcich snímok pokutového kopu [13]

Videa obsahujú aj ďalšie informácie okrem tých, ktoré sú obsiahnuté v jednej statickej snímke videa. Dodatočne obsahujú časovú zložku, ktorá poskytuje dôležité informácie o pohybe. V niektorých prípadoch je možné rozoznať určitú činnosť z jednej snímky. Na druhej strane pri ďalších činnostiach môžu byť jednotlivé snímky nejednoznačné a práve časová zložka môže poskytnúť spôsob zachytenia zložitejšieho kontextu.

V posledných rokoch sa na riešenie problému rozpoznávania akcie z videa používajú modely konvolučných neurónových sietí, ktoré poskytujú najlepšie výsledky v úlohách ako sú klasifikácia, rozpoznávanie, segmentácia a detekcia objektov. Dôležitým rozdielom medzi obrázkami a videom je časová informácia, ktorá je serializovaná do sekvencie snímok. V praxi sa často využíva kombinácia konvolučných a rekurentných neurónových sietí.

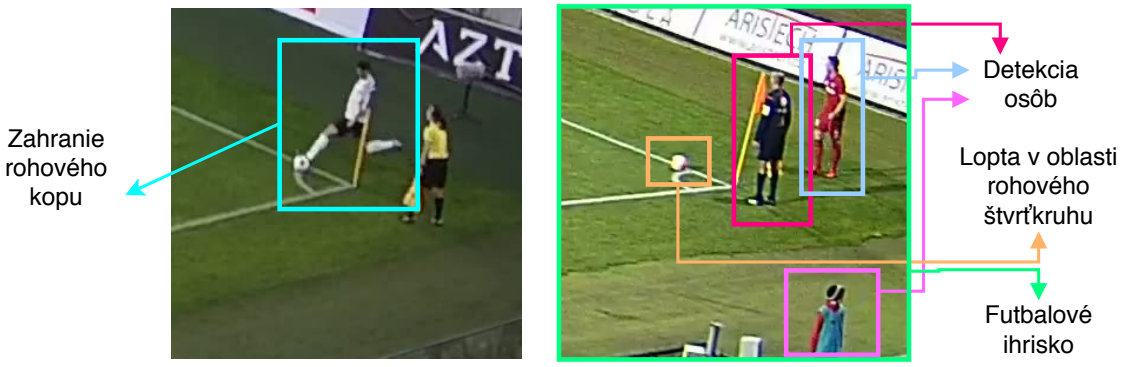
Dôležitým faktorom rozhodujúcim vo vývoji konvolučných sietí bol pokrok vo

výpočtovom výkone GPU, ktorý viedol k nárastu veľkosti siete na milióny parametrov.

Napriek mnohým výskumným prácam a veľkému pokroku v tejto oblasti zostalo mnoho problémov nevyriešených. Medzi hlavne problémy v oblasti rozpoznávania akcie z videa patria:

1. **Obrovské výpočtové náklady** – pri jednoduchej 2D konvolučnej sieti na klasifikáciu 101 tried má táto sieť približne 5 miliónov parametrov. Avšak rovnaká architektúra pri použití 3D štruktúry vedie až k približne 33 miliónom parametrom. Tento ich vysoký počet parametrov a zároveň veľký objem dát má za následok dlhšiu dobu natrénovania siete – niekoľko dní až mesiacov. Okrem toho toto rozsiahle vybavenie architektúry vyžaduje použitie výkonného a drahého vybavenia (*hardware*).
2. **Zachytenie zložitejšieho kontextu** – rozpoznanie akcie zahŕňa zachytenie časopriestorového kontextu v rámci snímok. Niekedy však samotná detekcia priestorových objektov nestačí, pretože pohybové informácie majú detailnejšiu charakteristiku. Napríklad video s plavcom v bazéne, ktorý pláva rôznym spôsobom (prsia, kraul). Klasifikátor vie identifikovať ľudské telo, bazén, ale charakter periodického pôsobenia štýlu, akým plavec pláva, je už komplexnejší.
3. **Štandardizované trénovacie dáta** – z praktického hľadiska je v súčastnosti veľmi zložité manuálne zozbierať, anotovať a uložiť videá. Z toho dôvodu sa často využívajú už existujúce súbory dát. Medzi najviac používané patria *UCF101* a *Sports1M*.

Jedna z hlavných techník rozpoznávania akcie je počítačové videnie, konkrétne rozpoznávanie činnosti na základe videa. Vizuálna časť videa poskytuje základné informácie o udalostiach alebo akciách vo videu. Existuje niekoľko techník, ktoré sa v poslednej dobe začali využívať na rozpoznávania videa. V ďalšej časti práce sa bude venovať pozornosť rôznym typom architektúr na rozpoznávanie akcie.



Obr. 2.11: Detekcia objektov z vizuálnej časti videa

2.4.1 Jednoprúdové konvolučné neurónové siete

Na porozumenie obsahu multimediálnych dát sú využívané konvolučné neurónové siete (*Convolutional Neural Network* – CNN), ktoré poskytujú najlepšie výsledky v oblasti rozpoznávania, segmentácie a detekcie rôznych udalostí. Kľúčovými faktormi, ktoré viedli k lepším výsledkom a podporili proces učenia, boli techniky rozšírenia siete na desiatky miliónov parametrov a použitie rozsiahlejších dát.

Pri študovaní konvolučnej neurónovej siete aplikovanej na rozsiahlu klasifikáciu videa sa zistilo, že okrem informácií o objektoch v obraze je účinne analyzovať ich komplexný časový vývoj. Z hľadiska modelovania takejto siete je dôležité preskúmať nasledujúce otázky. Aký časový model v architektúre CNN najlepšie využije informácie o lokálnom pohybe prítomnom vo videu? Do akej miery tento proces zlepšuje výkon takejto siete?

Z výpočtového hľadiska si CNN siete vyžadujú značne dlhé obdobie tréningu, aby účinné optimalizovali milióny parametrov. Tento problém sa ďalej znásobuje pri rozširovaní architektúry v čase, pretože sieť musí spracovať nielen obraz, ale aj niekoľko snímok videa súčasne.

Efektívny prístup na urýchlenie takéhoto systému CNN vyžaduje dva samostatné prúdy spracovania:

1. **Kontextový prúd** (*Context Stream*) – ktorý sa učí funkcie na snímkach s nízkym rozlíšením.
2. **„Fovea“ prúd s vysokým rozlíšením** – ktorý pracuje v strednej oblasti snímky.

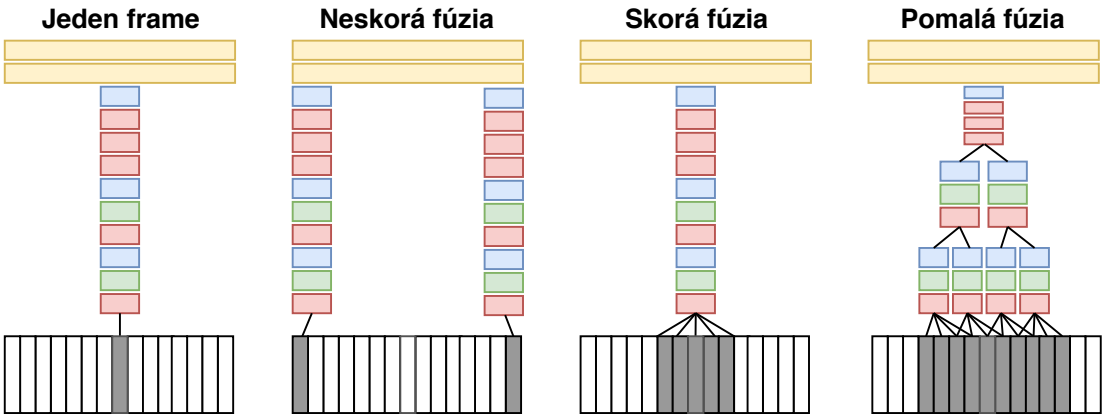
S použitím tejto architektúry sa zvýši výkonnosť siete dvojnásobne až štvornásobne – a to z dôvodu zníženia rozmerov vstupu, pričom bude zachovaná presnosť klasifikácie.

Každý videoklip je zložený z určitého počtu súvislých snímok v čase, a preto je možné naučiť sieť časopriestorové funkcie. Detailnejšie vysvetlenie spájania informácií naprieč časovej zložke bude vysvetlené na nasledujúcich príkladoch. Najprv bude vysvetlená základná jednosnímková fúzia, a potom jej rozšírenie v čase podľa odlišných typov fúzie:

- **Jednosnímková fúzia** (*Single Frame*) – je základná jednosnímková architektúra, používajúca statický obraz na klasifikácii. Vstupom do takejto siete je snímka s rozmermi $170 \times 170 \times 3$ pixelov. Výsledná architektúra s použitím skrátenej notácie vyzerá nasledovne: *K(96,11,3)-N-Po-K(256,5,1)-N-Po-K(384,3,1)-K(384,3,1)-K(256,3,1)-Po-PP(4096)-PP(4096)*, kde *K(d,f,s)* predstavuje konvolučnú vrstvu s d filtrami s priestorovou veľkosťou $f \times f$, ktorá sa aplikuje na vstup s krokom s. *PP(n)* je plne prepojená vrstva s n uzlami. *Po* predstavuje *Pooling* vrstvu a *N* špecifikuje normalizované vrstvy. Posledná

vrstva je pripojená k *softmax* klasifikátoru s hustým (*dense*) pripojením.

- **Skorá fúzia** (*Early Fusion*) – toto rozšírenie fúzie kombinuje informácie z celého časového okna okamžite na úroveň pixelov. Samotná implementácia prebieha pomocou úpravy filtrov prvej konvolučnej vrstvy modelu jednosnímkovej fúzie, a to rozšírením na veľkosť $11 \times 11 \times 3 \times T$ pixelov, kde T je určitý časový rozsah. Toto včasné a priame pripojenie umožňuje sieti zistiť lokálny smer a rýchlosť pohybu.
- **Neskorá fúzia** (*Late Fusion*) – je vytvorená z dvoch samostatných jednosnímkových sietí so zdieľaným parametrom, ktoré sú od seba vzdialené 15 snímok. Následne sa tieto dva prúdy spoja a vytvoria prvú plne prepojenú vrstvu, ktorá vie vypočítať globálne charakteristiky pohybu porovnaním výstupov oboch prúdov.
- **Pomalá fúzia** (*Slow Fusion*) je vyvážený typ fúzie, ktorý je vytvorený kombináciou skorej a neskorej fúzie. Tie pomaly spájajú časove informácie v celej sieti, takže vyššie vrstvy získavajú prístup k postupným globálnym informáciám jak v priestorovej, tak aj časovej dimenzii. Tento spôsob fúzie je možné implementovať rozšírením pripojenia všetkých konvolučných vrstiev v čase a vykonávaním časových konvolúcií.



Obr. 2.12: Prístupy fúzie časovej dimenzie prostredníctvom siete

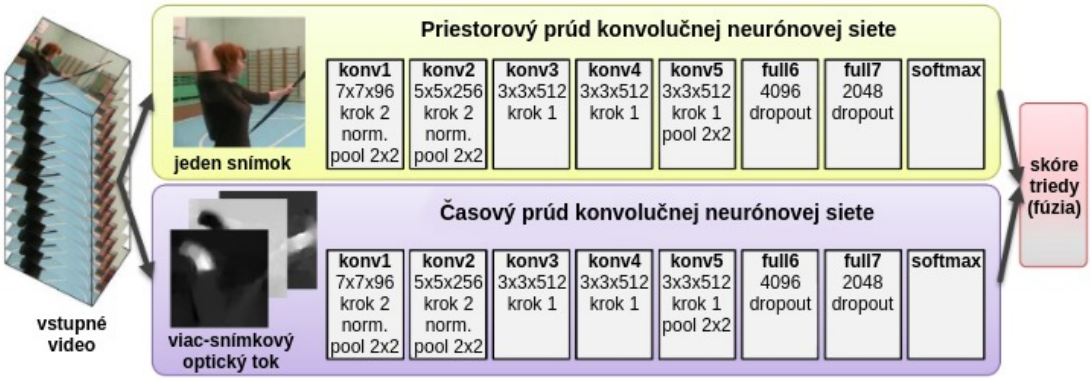
Ako je vidieť na obr. 2.12, vstup do všetkých fúzií tvoria za sebou idúce snímky. Pri **jednosnímkovej fúzii** sa spájajú informácie zo všetkých snímok až v poslednom kroku. **Neskorá fúzia** využíva dve siete, so zdieľanými parametrami, ktoré sú od seba vzdialené 15 snímok, pričom na konci skombinuje ich predikcie. Kombinácia **skorej fúzie** prebieha v prvej vrstve konvolúciou naprieč desiatimi snímkami. **Pomalá fúzia** zahŕňa fúziu s viacerými fázami. Je to „balance“ medzi skorou a neskorou fúziou [17].

2.4.2 Dvojprúdové konvolučné neurónové siete

Architektúra tohto typu sietí vychádza z podobnosti s ľudskou častou mozgu – *visual cortex*. Tu sa nachádzajú dve časti – ventrálny prúd, ktorý vykonáva rozpoznávanie objektov, a dorzálny prúd, ktorý rozpoznáva pohyb.

Na Oxfordskej univerzite v roku 2014 *K. Simonyan* a *A. Zisserman* prišli s novou architektúrou, ktorá bola postavená na jednokonvolučných neurónových sieťach, pričom autori stavajú na jej nedostatkoch. Vzhľadom na náročnosť naučiť neurónovú sieť pohybové prvky bola vytvorená vrstva vektorov optického toku. Namiesto zaužívanej jednej siete pre priestorový kontext má táto nová architektúra dve samostatné siete (obr. 2.13) – jednu pre **priestorový kontext** a jednu pre **kontext pohybu**.

Video je možné rozložiť na priestorovú a časovú zložku. **Priestorová časť** zahŕňa individuálnu snímku, kde je zachytená určitá informácia o prostredí a objektoch vo videu. **Časová zložka** definuje formu pohybu, ktorý vzniká medzi jednotlivými snímkami. Na základe týchto častí je navrhnutá architektúra konvolučnej neurónovej siete, ktorá sprostredkúva rozpoznávanie akcie z videa. Na obr. 2.13 je znázornená implementácia do dvoch prúdov – priestorového a časového. Oba prúdy sú navrhnuté pomocou hlbokej konvolučnej siete a výsledok z oboch časti *softmax* je kombinovaný s neskorou fúziou (posledná časť oboch prúdov).

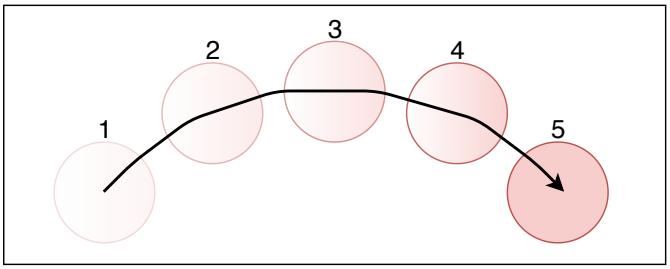


Obr. 2.13: Architektúra dvojprúdovej konvolučnej neurónovej siete [18]

1. **Priestorový prúd** (*Spatial Stream*) rozpoznávania konvolučnej neurónovej siete analyzuje jednotlivé videosnímky, čím efektívne rozpoznáva akcie zo statických obrázkov. Samotný statický obraz je v tomto ohľade dôležitý, pretože niektoré akcie sú úzko prepojené s konkrétnymi objektmi.
2. **Časový prúd** (*Temporal Stream*) rozpoznávania konvolučnej neurónovej siete je vytvorený niekoľkými zložkami optického toku po sebe idúcich snímok. Takýto vstup popisuje pohyb medzi videosnímkami, čo uľahčuje rozpoznávanie.

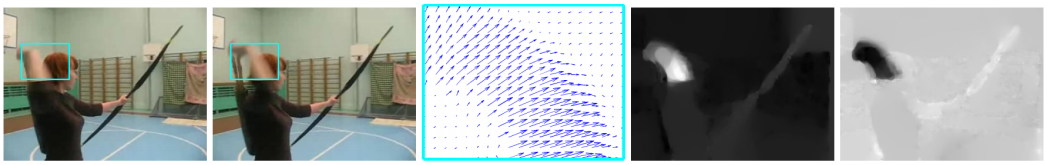
2.4.3 Optický tok

Optický tok (*Optical Flow*) sa dá definovať ako viditeľný pohyb objektu v obraze a zdanlivý „tok“ pixelov v obraze. Je to výsledok premietania 3D pohybu do dvojrozmernej obrazovej roviny. Optický tok sa využíva hlavne v oblasti sledovania objektov. Výskumné štúdie *Lucas-Kanade* [15] a *Horn-Schuncka* [16] poskytujú veľmi podrobný a zrozumiteľný opis optického toku, jeho výpočet a oblasti použitia.



Obr. 2.14: Vektor optického toku pohybujúceho sa objektu

Na obr. 2.15 v časti (a),(b) je znázornený pár po sebe idúcich videosnímok s oblasťou pohybujúcej sa ruky, ktorý je vyznačený azúrovým obdĺžnikom. Ďalší obrázok (c) upresňuje detailnejší záber hustého (*dense*) optického toku vo vyznačenej oblasti. (d): horizontálna zložka závislá na vektore posunu (vyššia intenzita odpovedá kladným hodnotám, nižšia intenzita záporným hodnotám). (e): vertikálna zložka. Obrázok (d) a (e) zvýrazňujú pohybujúcu sa ruku a luk.



Obr. 2.15: Princíp optického toku [18]

V tejto časti bude detailnejšie vysvetlená architektúra vstupu optického toku. Hustý (*dense*) optický tok tvorí súbor vektorov d_t medzi pármi nasledujúcich snímok t a $t+1$. Pomocou d_t je označený vektor posunu bodov (u,v) snímky t, ktorý vytvára posun medzi bodom snímky t a zodpovedajúcim bodom v nasledujúcej snímke $t+1$. Horizontálna a vertikálna zložka vektorového poľa je označená ako d_t^x a d_t^y, viac obr. 2.15: (d),(e). Naskladaním týchto vektorov z L nasledujúcich snímok vznikne tzv. prietokový kanál $d_t^{x,y}$, ktorý charakterizuje pohybovú zložku a vytvára celkovo $2L$ vstupných kanálov [18].

3 Predspracovanie vstupných dát

V tejto kapitole bude detailnejšie vysvetlená najdôležitejšia a hlavná časť tejto práce: **analýza** a **predspracovanie** vstupných dát. Práve kvalita vstupných dát zohráva hlavnú úlohu pri vytváraní účinného klasifikátora neurónovej siete. Prvým krokom na dosiahnutie tohto cieľa bolo analyzovať a pochopiť sémantický obsah vstupných dát.

Diplomová práca bola zameraná na detekciu rohového kopu a z toho dôvodu bolo dôležite nájsť spôsob, akým spracovať vstupné dáta na detekciu tejto udalosti. Podstatné bolo vybrať správny kontext z obsahu videozáznamu, v ktorom sa nachádzajú jedinečné príznaky tejto udalosti. Detailnejší popis vstupných dát bude vysvetlený v časti 3.1 Pri samotnej realizácii boli navrhnuté a následne spracované dva spôsoby vytvorenia vstupných dát, podľa zachytávaného kontextu. Tento kontext je možné rozdeliť na dve časti.

1. **Statický kontext** – táto časť bola zameraná na statickú analýzu obrazu pri procese zahrania rohového kopu. Pri analýze bola zvolená jedinečná vlastnosť tejto udalosti – lopta v oblasti rohového štvrťkruhu. Z toho dôvodu samotná detekcia udalosti spočívala v rozpoznávaní lopty na konkrétnom mieste.
2. **Dynamický kontext** – tento spôsob zahŕňa zachytenie pohybovej zložky v oblasti rohového štvrťkruhu. Konkrétne sa jedná o celý proces zahrania rohového kopu: rozbeh hráča smerom k lopte, samotné zahranie rohového kopu a následne opustenie hráča tejto oblasti.

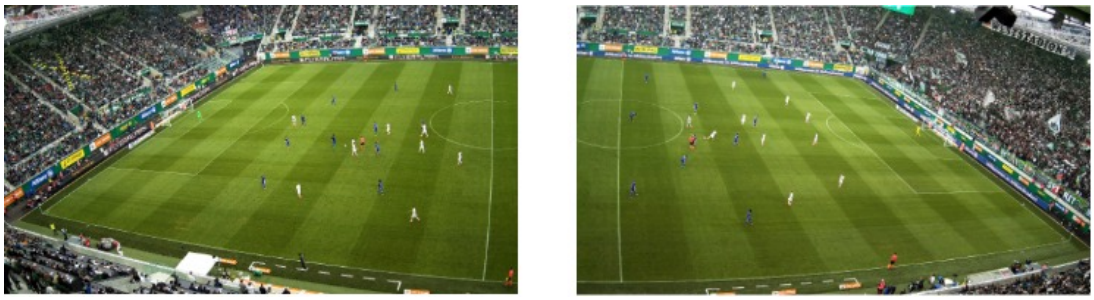
Samotná úloha predspracovania vstupných dát sa skladá z troch hlavných častí:

1. **Orezanie** snímok a **zmena FPS** (*Frames Per Seconds*). Obe tieto časti – výrez a zmena snímkovej frekvencie, boli spracované v jednom kroku kvôli optimálnej rýchlosti úpravy týchto videoklipov.
2. **Segmentácia videozáznamov** s presne definovanou dĺžkou periódy (napr. 2 sekundy). Ako vstupné dáta do tohto procesu boli použité výrezy z prvého kroku.
3. **Extrakcia snímok** z video sekvencie alebo použitie **metódy akumulácie**. V tejto časti z vytvorených segmentov extrahovali snímky, ktoré boli použité ako vstup do neurónovej siete.

Celý proces predspracovania vstupných dát bude detailnejšie rozobratý v nasledujúcich sekciách tejto kapitoly. Pri praktickej realizácii bol vytvorený automatizovaný algoritmus, ktorý využíva multimediálny softvér – ***FFmpeg***, umožňujúci editáciu videozáznamov.

3.1 Analýza vstupných dát

Vstupné dáta pre túto prácu boli poskytnuté spoločnosťou CamVision. Táto spoločnosť pochádza z Česka a zaoberá sa vývojom softvéru na analýzu futbalových zápasov. Dohromady bolo zozbieraných 40 videozáznamov, pričom každý videoklip obsahuje približne polovicu zápasu (jeden polčas). Celková dĺžka videozáznamov predstavuje 35 hodín a 23 minút. Veľkosť vstupných dát sa dostala na úroveň 320 Gb. Každý videozáznam obsahuje záber z ľavej a pravej kamery, pričom každá časť zachytáva danú polovicu futbalového ihriska (obr. 3.1). Kvalita jednotlivých videozáznamov je rôzna.



Obr. 3.1: Ukážka hracej plochy zo snímky videozáznamu

Pri analýze vstupných dát bolo zistených niekoľko aspektov, ktoré môžu pozitívne, ale aj negatívne ovplyvniť trénovanie neurónovej siete:

- niektoré videá **neboli** nahraté **v 4K rozlíšení**,
- **uhol natočenia** kamery bol odlišný pri rôznych videoklipoch,
- **podmienky hry** – dážď, hmla, hmyz v objektíve kamery, trasúca sa kamera alebo rôzna veľkosť štadióna,
- **spôsob** zahrania **rohového kopu** – napríklad zahranie „na krátko“, keď hráč zahrievajúci rohový kop prihrá najbližšiemu spoluhráčovi,
- **malý počet** anotovaných **udalostí** rohových kopov.

Ďalším krokom k príprave vstupných dát je vytvorenie anotovanej databázy video sekvencií. Vzhľadom na to, že realizovaný typ neurónovej siete patrí do kategórie učenia s učiteľom, je potrebné manuálne vytvoriť vstupné sekvencie videozáznamov. Preto jediná možnosť, ako vytvoriť súbor anotácií je prezretie každého videozáznamu a označenie udalostí a prislúchajúceho času. V diplomovej práci bol anotovaný jeden typ udalosti – **rohový kop**. Po vytvorení takejto databázy je ďalším krokom predspracovanie vstupných dát.

3.2 Orezanie oblasti z video sekvencií

Prvým krokom k vytvoreniu vstupných dát bolo orezanie originálnych videoklipov na výrez, zodpovedajúci danej udalosti. Vzhľadom na to, že neurónová sieť má určité obmedzenia k veľkosti vstupu, bolo vytvorených niekoľko výrezov s rôznymi veľkosťami. Na druhej strane, veľkosť výrezu musela byť dostatočne veľká, aby bola zachytená sémantická časť rohového kopu. Zvolené rozmery, ktoré boli použité na trénovanie neurónovej siete, boli: 150×150 a 299×299 pixelov. Ak bol rozmer vstupného obrazu väčší ako požadovaný rozmer na vstup do neurónovej siete, tak bol podvzorkovaný do príslušnej hodnoty.

Pred procesom orezávania videoklipov bolo potrebné nájsť počiatočný bod, ktorý slúžil ako záchytný bod. Na obrázku 3.2 je možné pozorovať, že snímok je vycentrovaný k vrcholu rohového štvrťkruhu. Súradnice tohto vrcholu boli nájdené pomocou online softvéru (*Image Map Generator*). Následne boli nájdené súradnice ľavej hornej časti vyrezávaného útvaru. Nakoniec je potrebné definovať rozmery výsledného výrezu.

Príklad zápisu vytvorenia modelovania vyrezávaného obrazu je znázornený v rovnici 3.1, pričom $\boldsymbol{v}$ predstavuje zápis výsledného výrezu, ktorý je použitý ako vstup do príkazového riadku procesu *FFmpeg*.

$$v = s : d : (x - o) : (y - o), \tag{3.1}$$

kde $\boldsymbol{s}$ a $\boldsymbol{d}$ sú rozmery výrezu a $\boldsymbol{x}$,$\boldsymbol{y}$ sú súradnice stredu výrezu, od ktorých je odpočítaná hodnota posunu $\boldsymbol{o}$ (*offset*).

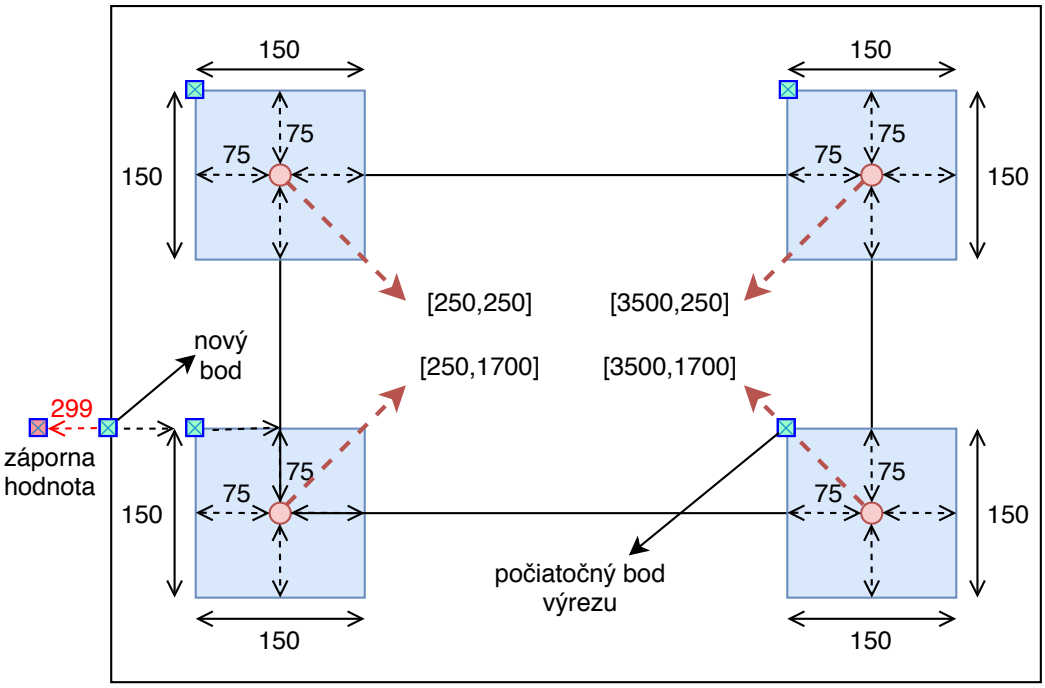
Obrázok 3.2 znázorňuje ukážky výrezu hracej plochy v oblasti udalosti rohového kopu.

Obr. 3.2: Ukážka výrezu videozáznamu s oblasťou rohového kopu

Pri samotnom orezávaní mohli nastať rôzne chybové udalosti, ktoré bolo potrebné patrične ošetriť. Napríklad mohla nastať situácia, keď pri definícií počiatočného bodu výrezu pretiekol rozmer do záporných hodnôt. Pri takto zostavenom výreze by došlo k vyvolaniu chybovej hlášky.

Na tento účel bol vytvorený algoritmus – **validátor súradníc**, ktorého funkciou bolo eliminovať pretečenie súradníc do záporných hodnôt. Algoritmus funguje tak, že v prípade zostavenia záporných súradníc súradnicovej osi, bude výsledná súradnica posunutá na počiatok prislúchajúcej osi sústavy súradníc.

Na obrázku 3.3 je schematicky znázornená hracia plocha s definovanou oblasťou výrezu (modrá oblasť). V schéme je zaznačená oblasť pretečenia súradníc do záporných hodnôt a následne nastavenie nového počiatočného bodu na začiatok súradnicovej osi.



Obr. 3.3: Proces validácie súradníc počiatočného bodu výrezu

V tom istom kroku ako sa vyrezáva oblasť videozáznamu, dochádza aj k zmene snímkovej frekvencie. Pôvodnú frekvenciu – 25 snímok/s je možné zredukovať na ľubovoľnú nižšiu hodnotu (napr. 5 snímok/s).

3.3 Segmentácia video sekvencií

V druhom kroku sa z vyrezaných oblasti video sekvencií vytvoria úseky (segmenty) videa s presne definovanou dĺžkou. V prípade segmentácie 45-minútového videa vznikne po segmentácii s dĺžkou segmentu 2 sekundy až 1350 krátkych videoklipov. Priemerný počet rohov na takto dlhý úsek je okolo 2-3 rohové kopy.

Na začiatku tohto procesu je potrebné zistiť dĺžku videozáznamu vyrezanej oblasti – to sa uskutočňuje pomocou príkazového riadka nástroja ***FFprobe***.

Samotná segmentácia je realizovaná jednoduchou iteráciou do hodnoty dĺžky videoklipu s definovaným krokom posunu (*offset*). Výpis 3.1 zobrazuje tri po sebe nasledujúce spracované segmenty, kde **-ss** je časová známka a **-t** je dĺžka segmentu.

Výpis 3.1: Proces segmentácie videoklipov

```
#!/bin/bash
$ ffmpeg -i vstup.mp4 -ss 00:00:00 -t 2 -c copy vystup.mp4
$ ffmpeg -i vstup.mp4 -ss 00:00:05 -t 2 -c copy vystup.mp4
$ ffmpeg -i vstup.mp4 -ss 00:00:10 -t 2 -c copy vystup.mp4
```

3.3.1 Pohybový detektor

Pri segmentácii vzniklo veľké množstve segmentov, ktorých obsah bol vo väčšine prípadov redundantný, tj. v oblasti rohového kopu sa nevykonáva žiaden adekvátny pohyb. Z toho dôvodu bolo potrebné tieto segmenty nejakým sofistikovaným spôsobom eliminovať a vyvážiť.

Na tento účel bol vytvorený **pohybový detektor** – je to skript, ktorý vypočíta percentuálne zastúpenie pohybu v danej video sekvencii. Následne sa vyberú najzaujímavejšie úseky, ako napríklad sekvencie, v ktorých bola detekovaná vyššia úroveň pohybu, alebo napríklad úseky s minimálnym pohybom. Výstupom je vyvážená dátová množina udalostí bez rohového kopu.

3.3.2 Generátor rohových kopov

K dokončeniu kompletnej dátovej sady už zostáva len vytvoriť záznamy obsahujúce rohové kopy. Na tento účelu bol implementovaný generátor rohových kopov. Na základe znalosti časových známok jednotlivých udalostí rohových kopov je možné tieto záznamy extrahovať z predspracovaného záznamu. Po dokončení tohto kroku vznikne kompletná dátová množina obsahujúca obe kategórie udalostí.

3.4 Extrakcia snímok z videosekvencie

Vzhľadom na to, že formát vstupu do neurónovej siete sú jednotlivé snímky z videoklipu, tak je potrebné tieto snímky extrahovať. Realizované boli dva prístupy:

- **Extrakcia snímok**
- **Akumulácia snímok**, ktorá bude detailnejšie vysvetlená v nasledujúcej sekcií.

V prípade štandardnej extrakcie individuálnych snímok ide o vytvorenie sady snímok pozostávajúcich z každého videoklipu. K realizácií tohto procesu bol opäť použitý multimediálny framework *FFmpeg*. Navrhnuté boli dve metódy extrakcie:

1. **Extrakcia s horizontálnym preklopením obrazu** – v tomto experimente bola použitá ľavá časť rohových kopov. Ak boli na vstup extraktoru aplikované sekvencie z pravej časti rohového kopu, tak boli horizontálne preklopené, aby korešpondovali s ľavou časťou (výpis 3.2).

Výpis 3.2: Horizontálne preklopenie videoklipu

```
#!/bin/bash
$ ffmpeg -i vstup.mp4 -vf hflip -c:a copy vystup.mp4
```

2. **Extrakcia bez použitia horizontálneho preklopenia obrazu** – v tomto prípade boli štandardne extrahované obe strany rohových kopov.

Po dokončení celého procesu extrakcie snímok z videoklipov sa automaticky vytvorí súbor vo formáte *.csv* s názvom: ***data_file.csv***. Súbor obsahuje všetky dôležité informácie o vstupných dátach:

- **názov videoklipu** (segmentu)
- **typ dátovej sady** (trénovacia alebo testovacia)
- **trieda kategórie** (rohový kop alebo ostatné)
- **počet extrahovaných snímok**.

Tab. 3.1: Ukážka dát súboru z výstupu extrahovania snímok

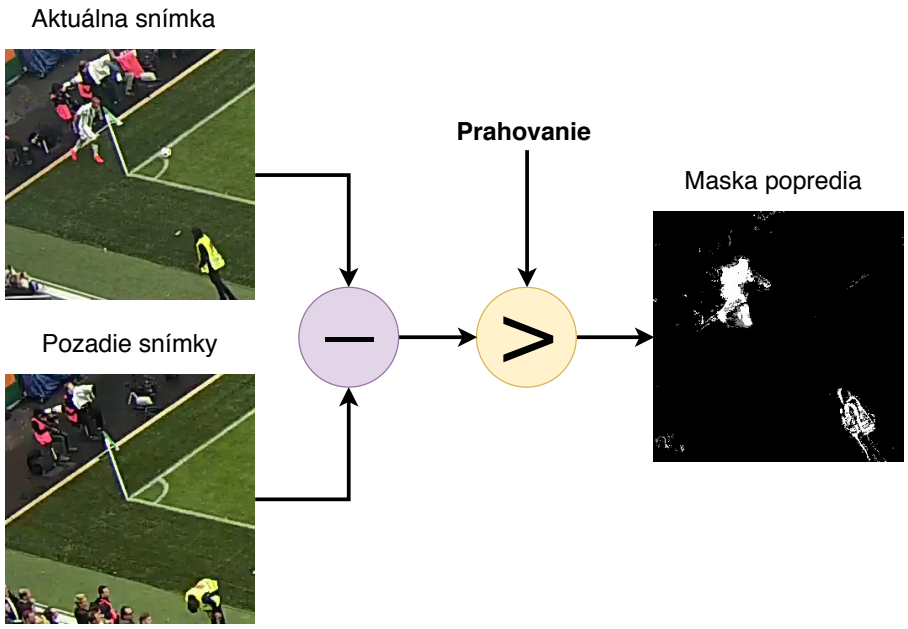
Typ sady	Trieda kategórie	Názov súboru	Počet snímkov
train	corners	24_China_Right_bottom_00:32:19	50
test	no_corners	9_Varšava_Right_bottom_00:42:51	50

Tento súbor je následne použitý pri kompilácii modelu neurónovej siete ako zdroj na načítanie trénovacej a testovacej dátovej sady procesu trénovania neurónovej siete.

3.4.1 Metóda akumulácie snímok

Druhý spôsob, ktorým je možné vytvoriť snímky na vstup do neurónovej siete, je akumulácia snímok. Princíp tejto metódy spočíva vo vytvorení jednej snímky (akumulovaná snímka), ktorá vznikne s použitím všetkých po sebe nasledujúcich snímok. Detailnejší popis vytvorenia takejto snímky je vysvetlený v nasledujúcich riadkoch. Implementácia algoritmu akumulácie snímok bola realizovaná prostredníctvom knižnice *OpenCV*.

1. **Odčítanie pozadia** (*Background Subtractor*) – použitie Gaussovského algoritmu na báze segmentácie pozadia a popredia. S pomocou tohto algoritmu je možné oddeliť pohybujúce sa objekty od statického pozadia. V prvom kroku sa vypočíta počiatočný model pozadia, zatiaľ čo v druhom kroku sa model aktualizuje, aby sa prispôsobil možným zmenám v scéne.



Obr. 3.4: Proces vytvorenia masky pomocou odčítania pozadia

2. **Akumulácia snímok** (*Accumulate*) – je to funkcia, ktorá umožňuje z video sekvencie zachytiť pohybujúce sa objekty a vyniesť ich do samostatnej snímky. Vzhľadom na to, že video je súbor po sebe idúcich snímok, tak riešením tejto metódy je akumulovať tieto snímky. Matematický zápis výpočtu akumulovanej snímky z video sekvencie popisuje rovnica 3.2:

$$a(x,y) \leftarrow a(x,y) + z(x,y) \quad \texttt{ak} \quad m(x,y) \neq 0, \tag{3.2}$$

kde $\boldsymbol{a}$ predstavuje akumulovanú snímku, $\boldsymbol{z}$ je zdrojová snímka a $\boldsymbol{m}$ je maska.

Výstupná snímka z procesu akumulácie zachytáva časovú zložku v rámci jednotlivých snímok videoklipu. Na obrázku 3.5 je zobrazená ukážka akumulovanej snímky.

Obr. 3.5: Výsledok akumulácie z 25 snímkov video sekvencie

Ukážka zdrojového kódu implementácie akumulácie snímok s použitím horintálneho preklopenia je zobrazená na výpise 3.3.

Výpis 3.3: Implementácia vytvorenia akumulovanej snímky

```
sub = cv2.createBackgroundSubtractorMOG2(128, cv2.THRESH_BINARY, 1)
sub.apply(frame)

avg_frame = np.float32(frame)
    while capture.isOpened():
        _, frame = capture.read()
        fg = sub.apply(frame)

        # calculate threshold value
        _, thresh = cv2.threshold(fg, 1, 1, cv2.THRESH_BINARY)

        # accumulate frame
        cv2.accumulate(frame, avg_frame, thresh)

        counter = counter + thresh

counter = cv2.merge((counter, counter, counter))
avg_frame = np.uint8(avg_frame / counter)

# horizontal flip
if flip:
  avg_frame = cv2.flip(avg_frame, 1)
```

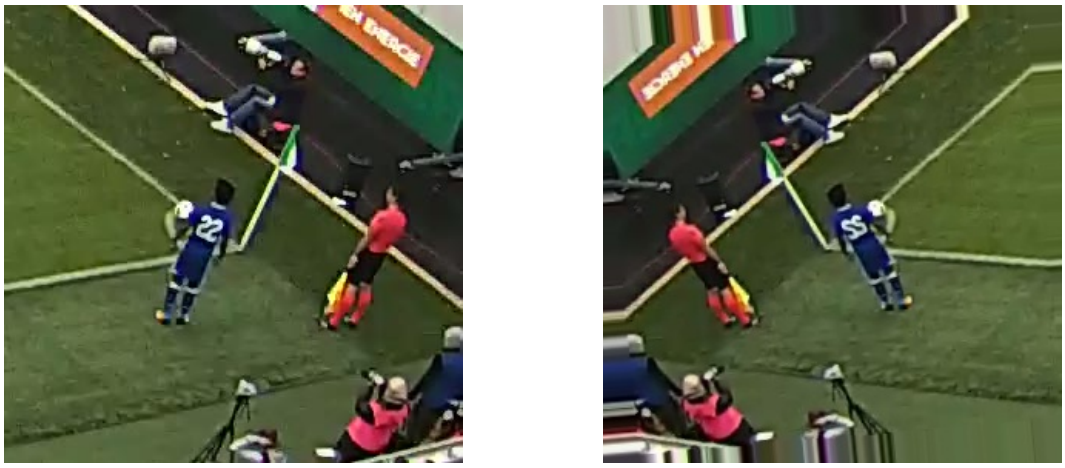
3.5 Augmentácia dát

Hlboké neurónové siete potrebujú veľké množstvo dát, aby dosiahli požadované výsledky. Na vytvorenie výkonného klasifikátora obrázkov pri veľmi malom počte trenovacích dát je potrebné nejakým spôsobom tieto dáta vygenerovať. Použitie určitých typov transformácií je v tomto ohľade dobrá možnosť ako zväčšiť rozsah dát prirodzenou metódou, pričom sa nezmení označenie triedy.

Na tento účel bol implementovaný generátor obrázkov, ktorý kombináciou viacnásobného spracovania umelo vytvára tréningové dáta. Vytvorenie takýchto modifikovaných verzií obrázkov je realizované napríklad náhodným otáčaním, posunom, orezaním alebo preklopením obrázka. V semestrálnej práci boli využité nasledujúce operácie:

- **normalizácia pixelov** – zmena rozsahu hodnôt pixlov z 1-255 na rozsah 0-1, čo vo výsledku urýchli trénovanie neurónovej siete.
- **horizontálne otočenie** – veľmi užitočná operácia, ktorá zrkadlovo preklopí ľavú časť rohových kopov na právu časť a opačne.
- **rotácia** – otočenie obrázka o náhodný počet stupňov v presne definovanom rozsahu.
- **priblíženie** – náhodné priblíženie alebo oddialenie snímky
- **šmyková transformácia** – náhodné posunutie každého bodu v stanovenom smere.

Trénovanie neurónovej siete na týchto rôznych variáciách obrázkov môže zlepšiť schopnosť zovšeobecnenia modelu. Implementácia tejto techniky bola spracovaná pomocou knižnice Keras. Na obrázku 3.6 je zobrazená ukážka augmentácie, pričom je vidieť otočenie snímky v horizontálnej rovine osi a šmykovú transformáciu.



Obr. 3.6: Vytvorenie novej snímky po procese augmentácie

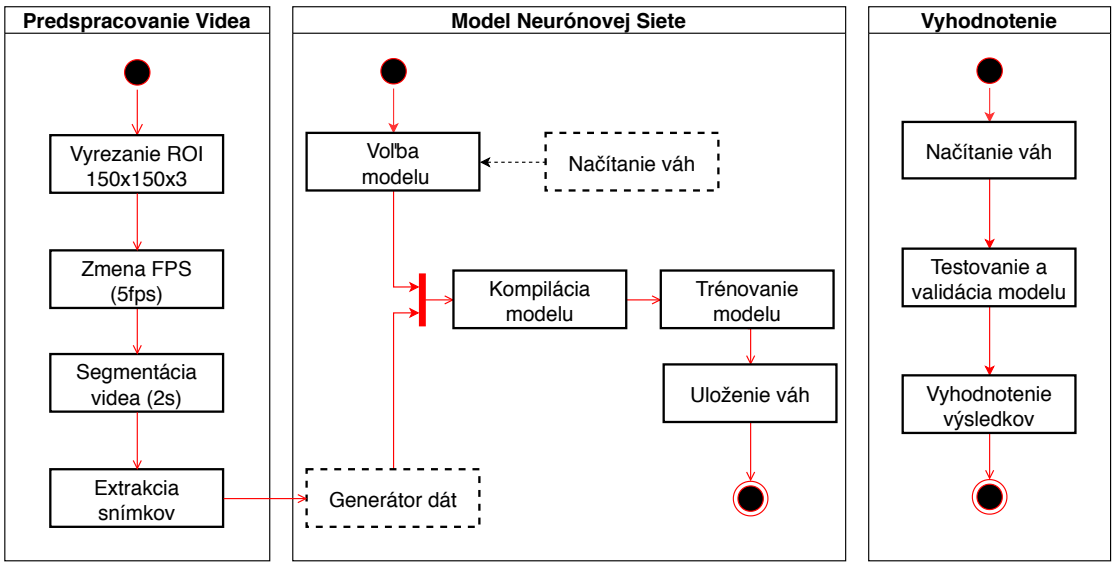
4 Návrh systému na detekciu udalostí

Účelom diplomovej práce bolo vybrať vhodnú metódu implementácie neurónových sietí na detekciu udalosti vo futbale a následne ju implementovať. Zvolená udalosť, ktorá bola detekovaná, bol rohový kop. V tejto kapitole je popísaný návrh a realizácia rôznych modelov neurónovej siete.

V prvej časti bude vysvetlený proces spracovania dát – získanie dát, anotácia a predspracovanie dát na vstup do neurónovej siete. Pre účely testovania bolo vytvorených viacero typov trénovacích a testovacích množín.

V ďalšej časti budú predstavené konkrétne implementácie algoritmov neurónovej siete. V prvotnej fáze práce bol na realizáciu vybratý model neurónovej siete od spoločnosti Google s názvom Inception. Motivácia a dôvod výberu modelu Inception spočívala v jeho komplexite a výsledkoch v úlohe rozpoznávania obrazu. Neskôr na porovnanie výsledkov boli spracované dva typy 3D konvolučnej neurónovej siete a rekurentná LSTM neurónová sieť. Implementácia neurónovej siete bola realizovaná v programovacom jazyku Python s použitím Tensorflow a platformy Docker.

V ďalšej kapitole práce budú zverejnené výsledky modelu neurónovej siete a vyhodnotené výsledky experimentov s použitím viacerých architektúr neurónovej siete. Výsledky budú vynesené do grafov a porovná sa úspešnosť jednotlivých modelov neurónovej siete. Na obrázku 4.1 je znázornený diagram procesu tvorby praktickej časti realizovanej v rámci diplomovej práce.



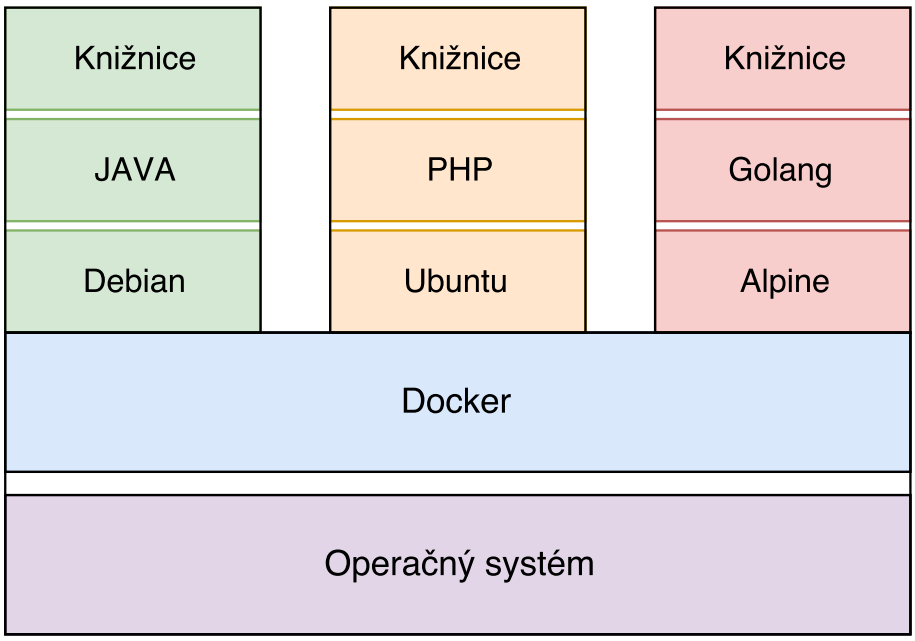
Obr. 4.1: Jednotlivé časti vytvoreného nástroja na detekciu udalosti

4.1 Zoznámenie sa s nástrojom Docker

Implementácie neurónových sietí so sebou prinášajú veľké množstvo knižníc, závislostí a podobne. Udržať takýto systém stabilný a aktuálny je veľmi náročná úloha, obzvlášť pri práci na iných zariadeniach. Každý počítač môže disponovať rozličným hardvérom, čo v niektorých prípadoch môže spôsobiť chyby pri inštalácii alebo spustení aplikácie. To isté platí aj pri použití iných typov operačných systémov ako napr. Linux, macOS alebo Windows.

Docker je nástroj určený na uľahčenie vytvárania, nasadzovania a spúšťania aplikácií pomocou kontajnerov. Cieľom je poskytnúť jednotné rozhranie pre aplikácie spustiteľné v ľubovoľnom operačnom systéme.

Kontajnery umožňujú vývojárovi zabaliť aplikáciu so všetkými potrebnými časťami, ako sú napríklad knižnice a iné dôležité súčasti aplikácie nevyhnutné na jej spustenie (viac obr. 4.2).



Obr. 4.2: Realizácia kontajnerov v nástroji Docker

Docker sa dá čiastočne prirovnať k virtuálnemu stroju. Na rozdiel od virtuálneho počítača však Docker namiesto vytvorenia celého virtuálneho operačného systému umožňuje aplikáciám používať to isté jadro ako systém, na ktorom bežia. Tým poskytuje výrazný nárast výkonu a znižuje veľkosť aplikácie.

Jedna z dôležitých vlastností tohto nástroja je, že je ***open source***. To znamená, že k nemu môže ktokoľvek prispieť a rozšíriť ho tak, aby vyhovoval jeho vlastným potrebám.

Praktická časť bola realizovaná pomocou tohto nástroja, čím sa podstatne zjednodušila možnosť spustiť vytvorenú aplikáciu na inom zariadení. Pomocou jediného príkazu je užívateľ schopný spustiť všetky služby z danej konfigurácie.

V práci boli vytvorené dva Dockefile súbory – jeden na prácu s videom a druhý k spusteniu aplikácie neurónovej siete. Na obrázku 4.3 je zobrazený Dockerfile použitý pre spustenie neurónovej siete. Prvý riadok začínajúci kľúčovým slovom **FROM** určuje nadradený obraz, z ktorého sa celý Dockerfile zostavuje.

Ďalšou dôležitou súčasťou tohto súboru je dokument s názvom **requirements.txt**, ktorý v sebe obsahuje všetky závislosti potrebné na skompilovanie a spustenie programu.

Dockerfile

```
FROM tensorflow/tensorflow:2.2.0rc2-gpu-py3

WORKDIR /app

# Install system packages
RUN apt-get update && apt-get install -y --no-install-recommends \
      bzip2 \
      g++ \
      git \
      graphviz \
      libgl1-mesa-glx \
      libhdf5-dev \
      openmpi-bin \
      wget && \
    rm -rf /var/lib/apt/lists/*

ADD requirements.txt requirements.txt
RUN pip install --upgrade pip
RUN pip install -r requirements.txt
RUN apt-get update
RUN apt-get install -y ffmpeg
COPY . .
ENV PYTHONPATH "${PYTHONPATH}:/app"
```

Obr. 4.3: Ukážka súboru Dockerfile s tensorflow

Obrovská výhoda spočíva v tom, že jednotlivé obrazy sú spustiteľné bez toho, aby sa v systéme inštalovali, takže je veľmi jednoduché ich neskôr zmazať. Okrem toho Docker zabezpečuje, že budú k dispozícii najnovšie verzie obrazov. To je dobrá vec z hľadiska bezpečnosti, pretože sa tým zabezpečí, aby sa nenainštaloval žiadny zraniteľný softvér.

4.2 Implementácia algoritmov neurónovej siete

Po úspešnom procese spracovania dát na vstup do neurónovej siete sa môže prejsť k samotnému modelu neurónovej siete. Na realizáciu detekcie udalosti bolo použitých niekoľko architektúr neurónovej siete (viac v ďalšej časti).

Aby sa predišlo pretrénovaniu modelu, bol k neurónovej sieti pridaný parameter včasného ukončenia (*Early Stopper*). Pozorovaná hodnota pri ukončení sa volá **validačná chyba** (*Validation Loss*). Ak sieť nezníži validačnú chybu v definovanom počte po sebe nasledujúcich etapách (prah ukončenia), celý proces trénovania sa ukončí. V podstate sa jedná o adaptáciu modelu na nové dáta (validačné dáta), ktoré model neurónovej siete predtým nevidel. Ak by sieť znižovala chybovosť len na trénovacích dátach, ale nie na validačných, tak by sa dostávala do nežiaduceho stavu – pretrénovanosti (*Overfitting*).

Ďalší užitočný nástroj pri trénovaní neurónovej siete je **TensorBoard**, ktorý poskytuje vizualizáciu výkonnosti siete. Umožňuje sledovanie metrík experimentu, ako je napr. chybovosť alebo presnosť siete pri validačných a trénovacích dátach. Pomocou nástroja TensorBoard sú výsledky trénovania automaticky vykreslené do grafov.

Jeden z dôležitých parametrov neurónovej siete je koeficient učenia, ktorý riadi veľkosť kroku optimalizátora smerom k minimalizácii straty. K lepšej konvergencii neurónovej siete na globálne minimum bol použitý parameter adaptívneho koeficientu učenia (*Adaptive Learning Rate*). V prípade, ak optimalizátor neznižuje validačnú stratu do definovanej prahovej hodnoty, koeficient učenia sa zredukuje o polovicu (factor = 0,5).

Pred samotným spustením trénovania modelu je potrebné rozdeliť dáta na dve separátne množiny – **trénovacie** a **testovacie** dáta. Trénovacie dáta sú použité na natrénovanie modelu a testovacie na jeho validáciu. Pomer rozdelenia týchto množín bol ustanovený na 80 % trénovacie a 20 % testovacie dáta. Samotný proces rozdelenia týchto množín je realizovaný pomocou naprogramovaného algoritmu, ktorý na vstup prijíma identifikátory konkretných štadiónov.

Ďalším krokom je výber metrík, ktoré budú dôležité pri analyzovaní a vyhodnocovaní výsledkov neurónovej siete. Zvolenou metrikou na analýzu neurónovej siete bola **presnosť klasifikácie**.

Implementácia modelov bola realizovaná za pomoci knižníc ***Tensorflow*** a ***Keras***. Následne sa vytvorený model skompiluje a spustí. Návod na inštaláciu prostredia, postup k samotnému spusteniu a validácii modelu neurónovej siete je priložený v prílohe A.

4.2.1 Inception model

Ako už bolo spomenuté v teoretickej časti, tento model pracuje s 2D obrazom a generuje predikcie na základe statickej informácie obsiahnutej v individuálnych snímkach. Veľkosť vstupu bola nastavená na hodnotu 150×150 pixelov.

V prvom kroku je potrebné model načítať. Model je predtrenovaný na *ImageNet* databáze, ktorá zahŕňa viac ako 14 miliónov obrázkov a 20 000 kategórií.

K modelu je následne pridaná vrstva spriemerovania **GAP** (*Global Average Pooling*), ktorá dopomáha k eliminácii pretrénovania neurónovej siete redukciou celkového počtu parametrov modelu. Vo výsledku ide o transformáciu 3D tenzoru na 2D maticu.

Ďalšiu pridanú vrstvu tvorí plne prepojená vrstva s aktivačnou funkciou **ReLU** (*Rectified Linear Unit*). Doplnkovou možnosťou je pridanie vrstvy **Dropout**, ktorá zlepší generalizovanie neurónovej siete. Na koniec je pripojená logistická vrstva s dvoma neurónmi a aktivačnou funkciou **softmax**, na účel klasifikácie dvoch tried.

Ako algoritmus minimalizácie chybovej funkcie (*Optimizer*) bol použitý **SGD** (*Stochastic Gradient Descent*). Oproti klasickému gradientu u SGD dôjde ku konvergencii k minimu skôr, pretože iterácia je realizovaná len pri určitom počte vzorov.

Výpis 4.1: Implementácia modelu neurónovej siete Inception

```
# create the base pre-trained model
base_model = InceptionV3(weights=weights, include_top=False)

# add a global spatial average pooling layer
x = base_model.output
x = GlobalAveragePooling2D()(x)

if dropout:
    print("INFO: using dropout")
    x = Dense(512, activation='relu')(x)
    x = Dropout(0.5)(x)
    x = Dense(512, activation='relu')(x)
    x = Dropout(0.5)(x)
else:
    x = Dense(1024, activation='relu')(x)

# and a logistic layer -- let's say we have 2 classes
x = Dense(len(data.classes), activation='softmax')(x)

# this is the model we will train
inception_model = Model(inputs=base_model.input, outputs=x)
```

4.2.2 3D konvolučná sieť

Pri implementácii 3D konvolučných neurónových sietí bol pôvodný rozmer 150×150 pixelov podvzorkovaný na veľkosť 80×80 pixelov z dôvodu pamäťového obmedzenia na grafickej karte. Vstup do tohto modelu je zostavený prostredníctvom sekvenčných snímok. Referenčná hodnota počtu snímok bola nastavená v rozmedzí 10-20 snímok. Vďaka prídavnému rozmeru je zachytená časová zložka danej sekvencie.

Ako algoritmus minimalizácie chyby pri tomto type konvolučných sietí sa uplatnil **RMSprop** (*Root Mean Square Propagation*), prípadne **Adam** (*Adaptive Moment Estimation*).

V praktickej časti boli implementované dva rozdielne typy 3D konvolučných neurónových sietí. Vo výpise 4.2 je zobrazená architektúra jedného z týchto modelov. Po konvolučných vrstvách bola k sieti pripojená plne prepojená vrstva s dvoma skrytými vrstvami. Medzi skrytými vrstvami bola implementovaná vrstva Dropout k lepšej regularizácií siete.

Výpis 4.2: Implementácia 3D konvolučnej neurónovej siete

```
"""
Build a 3D convolutional network, based loosely on C3D.
https://arxiv.org/pdf/1412.0767.pdf
"""
# Model.
model = Sequential()
model.add(Conv3D(
  32, (3, 3, 3), activation='relu', input_shape=self.input_shape
))
model.add(MaxPooling3D(pool_size=(1, 2, 2), strides=(1, 2, 2)))
model.add(Conv3D(64, (3, 3, 3), activation='relu'))
model.add(MaxPooling3D(pool_size=(1, 2, 2), strides=(1, 2, 2)))
model.add(Conv3D(128, (3, 3, 3), activation='relu'))
model.add(Conv3D(128, (3, 3, 3), activation='relu'))
model.add(MaxPooling3D(pool_size=(1, 2, 2), strides=(1, 2, 2)))
model.add(Conv3D(256, (2, 2, 2), activation='relu'))
model.add(Conv3D(256, (2, 2, 2), activation='relu'))
model.add(MaxPooling3D(pool_size=(1, 2, 2), strides=(1, 2, 2)))

model.add(Flatten())
model.add(Dense(1024))
model.add(Dropout(0.5))
model.add(Dense(1024))
model.add(Dropout(0.5))
model.add(Dense(self.nb_classes, activation='softmax'))
```

4.2.3 LSTM rekurentná sieť

Poslednou implementovanou architektúrou bola rekurentná neurónová sieť LSTM (*Long short-term memory*), ktorá má namiesto štandardných neurónových sieti spätnoväzbové pripojenia.

Na začiatku je potrebné extrahovať príznaky zo vstupnej snímky. Tieto príznaky sú získané pomocou kompilácie modelu Inception. To je možné docieliť odstránením plne prepojenej vrstvy na konci architektúry. Po inicializácii modelu je možné priviesť na vstup snímku, z ktorej sa vypočíta 2048 bitové pole hodnôt (výpis 4.3).

Výpis 4.3: Extrakcia príznakov pomocou Inception modelu

```
# Get model with pretrained weights.
base_model = InceptionV3(
    input_tensor=input_tensor,
    weights='imagenet',
    include_top=True
)

# We'll extract features at the final pool layer.
self.model = Model(
    inputs=base_model.input,
    outputs=base_model.get_layer('avg_pool').output
)
```

Vytvorené sekvencie sa automaticky ukladajú do príslušnej zložky, z ktorej sa následne načítajú do LSTM siete. Táto sieť je tvorená LSTM vrstvou, na ktorú sa pripája jedna skrytá vrstva s 512 neurónmi. Podobne ako to bolo pri predchadzajúcich architektúrach sa použila vrstva Dropout.

Výpis 4.4: Implementácia LSTM modelu neurónovej siete

```
"""
Build a simple LSTM network. We pass the extracted features from
 our CNN to   this model predomenently.
"""
# Model.
model = Sequential()
model.add(LSTM(2048, return_sequences=False,
               input_shape=self.input_shape,
               dropout=0.5))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(self.nb_classes, activation='softmax'))
```

5 Výsledky a experimenty

V tejto kapitole budú zverejnené výsledky realizácie praktickej časti práce na dvoch experimentoch. Prvým z nich je detekcia udalosti rohového kopu pomocou zachytenia statického obrazu, tj. individuálnych snímok. V tomto experimente sa za príznak udalosti rohového kopu považovala situácia, keď lopta vstúpila do oblasti rohového štvrťkruhu.

Druhý experiment, na rozdiel od prvého, zachytával časopriestorovú zložku dát, teda pohyb hráča smerom k lopte pri rozohrávke rohového kopu.

5.1 Príprava datasetov

Na oba experimenty bolo použitých 698 videoklipov. V tabuľke 5.1 je zaznačené rozdelenie dostupných dát do konkrétnych kategórií podľa udalosti.

Tab. 5.1: Distribúcia dát naprieč typu udalosti

Kategória udalosti	Celkový počet
Rohové kopy	84
Ostatné	614

Z tabuľky 5.1 vyplýva, že kategória rohových kopov obsahuje významne menšie množstvo udalostí ako kategória ostatné. K zabezpečeniu rovnomernej distribúcií dát, boli dáta z kategórie rohové kopy vygenerované formou posunu počiatku videoklipu o definovanú hodnotu posunu (*offset*). Týmto postupom bola okrem iného vytvorená jednoduchá augmentácia rohových kopov. Dáta v trénovacej sade boli vytvorene tak, aby testovacia sada neobsahovala udalostí rovnakého štadióna.

Hlavným obmedzením efektivity procesu trénovania bola malá vzorka vstupných dát. Z toho dôvodu sa pri vyhodnocovaní mohlo stať, že sieť by mala významne rozdielne výsledky pri inom rozdelení trénovacej a testovacej množiny dát. Na zvýšenie presnosti klasifikátora boli vytvorené tri unikátne datasety, ktorých výsledky sa na konci spriemerovali.

Výpis 5.1: Prikaz vytvorenia trénovacej a testovacej dátovej sady

```
#!/bin/bash
$ make data_organizer ARGS="--split --move -l 7 14 24 8 13"
```

Na rozdelenie stačí zadať unikátne identifikátory štadiónov, ktoré sa vzťahujú k testovacím dátam. Následne sa dáta presunú do separátnych zložiek – **test** a **train**.

5.2 Experiment č. 1: statická detekcia obrazu

Prvý vypracovaný experiment spočíval v detekcii rohových kopov z individuálnych snímok (*Image Recognition*). Na tento účel bola zvolená hlboká neurónová sieť s názvom **Inception**.

Princípom detekcie bolo zachytenie špecifických znakov pri rohovom kope. Príznakom tejto udalosti bola lopta v oblasti rohového štvrťkruhu. Výskyt lopty v tejto oblasti indikoval rohový kop.

Vstupom do takejto siete sú **individuálne snímky**, ktoré boli extrahované z predspracovaných videoklipov. Ako už bolo spomenuté v predchádzajúcej časti, kategórie udalostí obsahujú rozdielny počet videoklipov. K vytvoreniu rovnováhy medzi počtom udalostí v týchto kategóriách bola použitá technika zníženia snímkovej frekvencie pri kategórii s majoritným počtom dát.

Pri tomto experiment bola dôležitá hlavne veľká variabilita vstupných dát, ktorá bola ešte vylepšená prídavnou augmentáciou (sekcia 3.5). V ďalšej časti budú zobrazené výsledky experimentu.

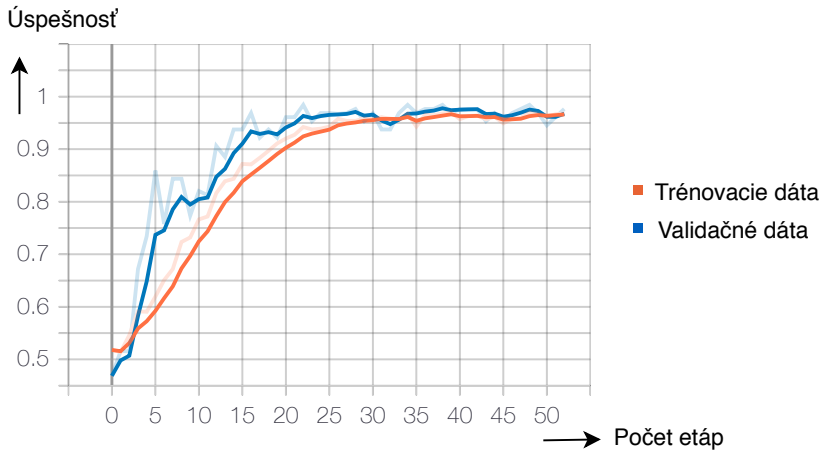
5.2.1 2D konvolúcia pomocou Inception modelu

V tabuľke 5.2 sú uvedené vlastnosti vstupu a hodnoty nastavenia neurónovej siete. Pri tomto experimente bol rozmer vstupu ustanovený na veľkosť 150×150 pixelov, čo predstavuje minimálne nastavenie rozmeru siete Inception.

Tab. 5.2: Hodnoty použitých parametrov na trénovanie modelu Inception

Názov parametra	**Hodnota parametra**
Snímková frekvencia	25 snímok/s
Rozmer snímky	150×150 pixelov
Rozmer vstupu	150×150 pixelov
Počet epoch	50
Počet validačných krokov	5
Batch	32
Vrstva Dropout	True
Optimalizátor	SGD(lr=0.0001, momentum=0.9)
Adaptívny koeficient učenia	True

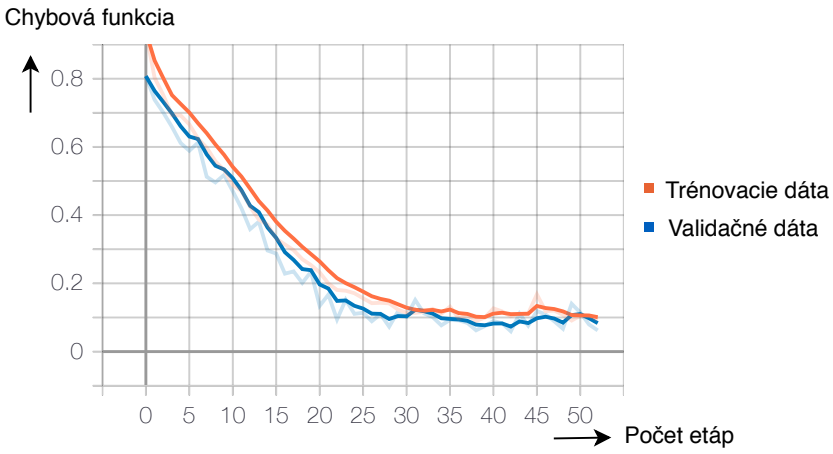
Na obr. 5.1 je zobrazený grafický priebeh úspešnosti trénovania neurónovej siete naprieč jednotlivými etapami. V grafe je možné pozorovať, že úspešnosť pri validačných dátach rastie rýchlejšie ako pri trénovacích. Tento jav je spôsobený pridanou vrstvou Dropout, ktorá spôsobuje deaktiváciu neurónov pri vyhodnocovaní danej etapy (nie je to nežiaduci stav).



Obr. 5.1: Priebeh trénovania modelu Inception

Obrázok 5.2 znázorňuje graf znižovania chyby (*Loss Function*) pri klasifikovaní dát počas procesu trénovania. Táto chybová funkcia vyjadruje mieru schopnosti siete adaptovať sa na nové (validačné) dáta.

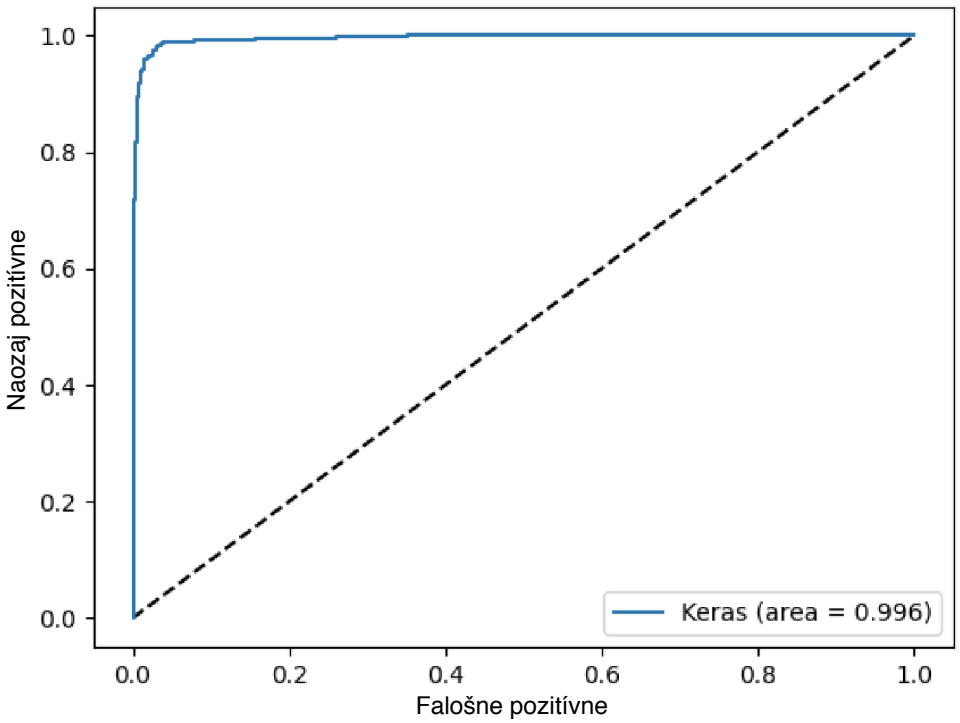
Celý proces trénovania sa skončil po 50. etape, pričom validačná chyba vykonvergovala k minimu už pri 40. etape.



Obr. 5.2: Priebeh konvergencie validačnej chyby modelu Inception

K vytvoreniu kvalitného klasifikátoru bolo potrebné zvážiť použitie metriky **Precíznosť** (Precision), podľa ktorej je možné určiť, ako sieť dokáže eliminovať falošné predpovede rohových kopov.

Nakoniec bola vypočítaná **ROC krivka**, ktorá slúži k optimalizácií binárneho klasifikátoru v závislosti na nastavení klasifikačného prahu (obr. 5.3). V pravom dolnom rohu je vypočítaná hodnota **AUC** (*Area Under the Curve*), ktorá predstavuje mieru výkonnosti v rámci všetkých možných prahov. Čím vyššia hodnota, tým lepšie sieť dokáže klasifikovať jednotlivé udalosti.



Obr. 5.3: Graf ROC krivky

Pomocou ROC krivky je možné nájsť **optimálny prahový bod** (*Threshold*). V tejto práci bol použitý **Youdenov index**, ktorý maximalizuje vertikálnu vzdialenosť od línie rovnosti k bodu [x,y]. Inými slovami, Youndenov index J je bod na krivke ROC, ktorý je najvzdialenejší od línie rovnosti (diagonálna línia). Hlavným cieľom je maximalizovať rozdiel medzi **TPR** (*True Positive Rate*) a **FPR** (*False Positive Rate*). Rovnica 5.1 vyjadruje výpočet prahového bodu pomocou Youdenovho indexu:

$$d = \sqrt{(1 - S_n)^2 + (1 - S_p)^2}, \tag{5.1}$$

kde S_n je senzitivita a S_p vyjadruje špecifickosť.

5.3 Experiment č. 2: zachytenie pohybovej zložky

V rámci druhého experimentu bola detekovaná **časopriestorová zložka** dát. Princípom tohto experimentu bolo zachytenie pohybu hráča pri rozohrávke rohového kopu.

Najväčším nedostatkom tohto experimentu bola malá množina vstupných dát. Z toho dôvodu boli nové dáta vygenerované za pomoci definovania posunu začiatku, vďaka čomu mala sieť hladší priebeh konvergencie k minimu.

Do experimentu boli aplikované dve architektúry 3D konvolučných neurónových sietí a rekurentná LSTM neurónová sieť. V nasledujúcej časti budú zobrazené výsledky experimentu, pri každej architektúre zvlášť.

5.3.1 3D konvolučné neurónové siete

V tab. 5.3 sú uvedené hodnoty parametrov použitých na trénovanie modelu neurónovej siete. **Dĺžka sekvencie** v tomto prípade predstavuje ďalší rozmer, v ktorom sú jednotlivé snímky zachytávajúce pohybovú zložku. Ustanovená hodnota tohto parametra predstavuje 20 snímok. Preto pôvodný videoklip, obsahujúci 50 snímok, bol podvzorkovaný do požadovanej veľkosti.

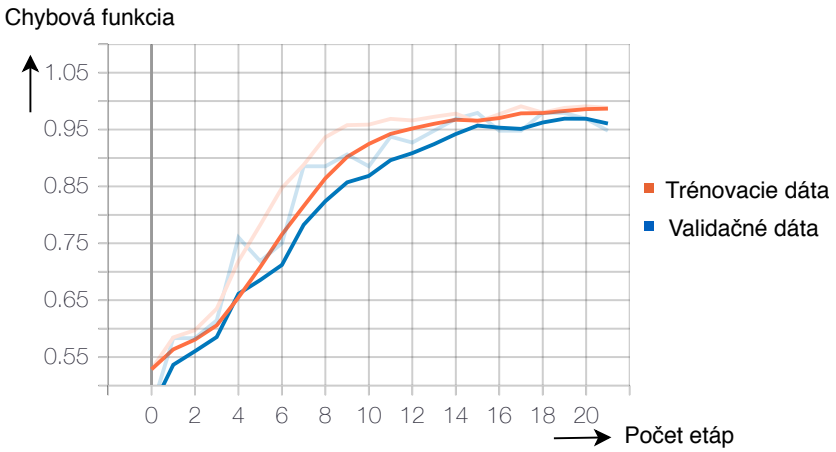
Rozmer výrezu, v ktorom bola zachytená pohybová zložka, bol rovnaký ako pri predchádzajúcom experimente – 150×150 pixelov. Na druhej strane, vzhľadom k vysokým pamäťovým nárokom 3D konvolučných neurónových sietí, bola definovaná veľkosť rozmeru vstupu do neurónovej siete znížená na 80×80 pixelov. Z toho dôvodu je nutné **podvzorkovať** pôvodný výrez na zvolenú hodnotu vstupného rozmeru.

Zvolený algoritmus minimalizácie chybovej funkcie bol Adam, ktorého konvergencia dosahovala najlepšie výsledky.

Tab. 5.3: Hodnoty použitých parametrov na trénovanie 3D konvolučnej siete

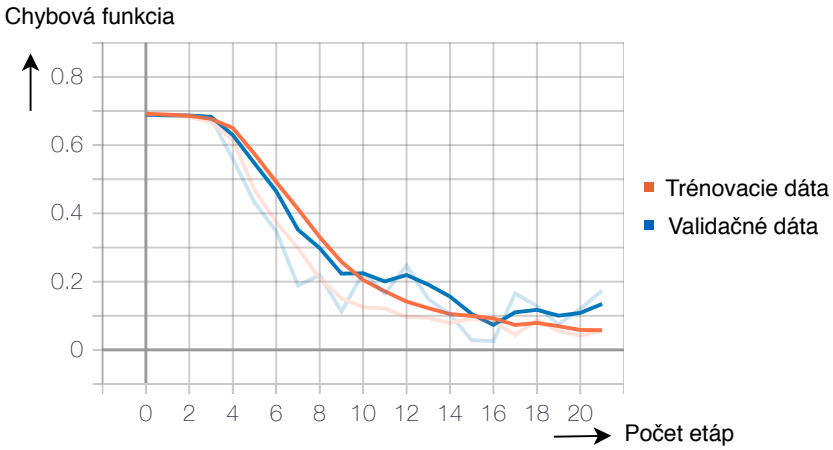
Názov parametra	**Hodnota parametra**
Dĺžka sekvencie	20 snímok
Snímková frekvencia	12 snímok/s
Rozmer snímky	150×150 pixelov
Rozmer vstupu	80×80 pixelov
Počet epoch	64
Počet validačných krokov	12
Batch	32
Vrstva Dropout	True
Optimalizátor	Adam

Obrázok 5.4 zobrazuje priebeh trénovania architektúry 3D konvolučnej neurónovej siete. Úspešnosť sa ustálila na hodnote približne 96 %. V grafe je možné pozorovať miernu fluktuáciu, ktorá bola spôsobená nižším počtom vstupných dát. Z toho dôvodu sa sieť nemusí dostatočné dobre naučiť rozlišovať niektoré podobné príznaky pri rozdielnych kategóriách.



Obr. 5.4: Priebeh trénovania modelu 3D konvolučnej siete

Podobná fluktuácia bola pozorovaná aj pri konvergencii k minimu u validačnej chyby. K minimalizácií chybovej funkcie došlo približne po 16. etape.



Obr. 5.5: Priebeh konvergencie chybovej funkcie modelu 3D konvolučnej siete

5.3.2 LSTM rekurentná neurónová sieť

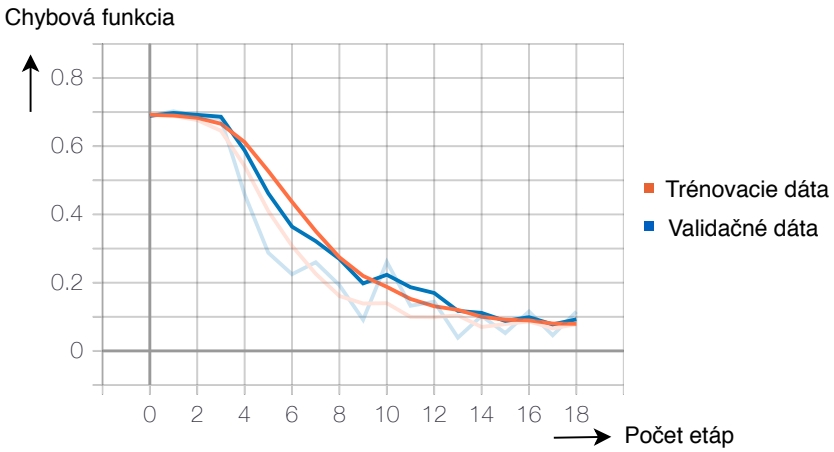
Poslednou architektúrou, ktorá bola v práci realizovaná, je rekurentná neurónová sieť LSTM. V tab. 5.4 sú špecifikované parametre nastavenia neurónovej siete.

Vďaka nízkej hardvérovej náročnosti bol zachovaný rovnaký pomer výrezu a vstupného rozmeru. Dĺžka sekvencie bola nastavená na hodnotu 25 snímok, čo predstavuje každú druhú snímku z videoklipu.

Tab. 5.4: Hodnoty použitých parametrov na trénovanie LSTM siete

Názov parametra	**Hodnota parametra**
Dĺžka sekvencie	25 snímok
Snímková frekvencia	25 snímok/s
Rozmer snímky	150 × 150 pixelov
Rozmer vstupu	150 × 150 pixelov
Počet epoch	64
Počet validačných krokov	12
Batch	32
Vrstva Dropout	True
Optimalizátor	Adam

Na obr. 5.6 je graf znázorňujúci priebeh trénovania neurónovej siete. Podobne ako pri 3D konvolučných neurónových sieťach bola pozorovaná mierna fluktuácia.



Obr. 5.6: Priebeh konvergencie chybovej funkcie modelu LSTM

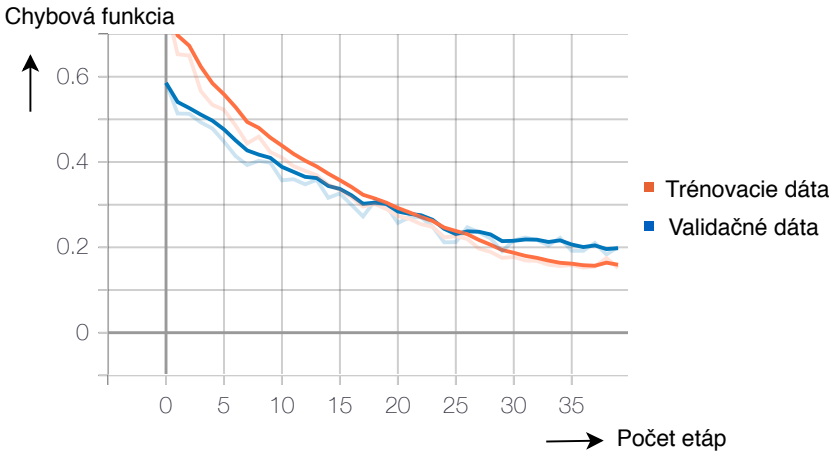
5.4 Experiment č. 3: akumulácia snímok

Posledný experiment, ktorý bol realizovaný v tejto práci, bola akumulácia snímok. Tento experiment je založený na rovnakom princípe ako experiment č. 1, rozpoznávanie statickej informácie z jedného snímku. Na rozdiel od prvého experimentu, v tomto experimente je vstup vytvorený akumuláciou snímok naprieč celej sekvencií.

Tab. 5.5: Hodnoty použitých parametrov na trénovanie LSTM siete

Názov parametru	**Hodnota parametru**
DÍžka sekvencie	25 snímkov
Snímková frekvencia	25 snímkov/s
Rozmer snímku	150 × 150 pixelov
Rozmer vstupu	150 × 150 pixelov
Počet epoch	50
Počet validačných krokov	5
Batch	32
Vrstva Dropout	True
Optimalizátor	SGD(lr=0.0001, momentum=0.9)

Na obr. 5.6 je graf znázorňujúci priebeh trénovania neurónovej siete, ktorý zobrazuje konvergenciu chybovej funkcie.



Obr. 5.7: Priebeh konvergencie chybovej funkcie pri akumulácií snímok

5.5 Porovnanie experimentov

Hlavným účelom týchto experimentov bolo interpretovať rôzne spôsoby prístupu k detekcií udalosti. Prvým experimentom bola detekcia individuálnych snímok a zachytenie statickej informácie. V druhom experimente bola zachytená pohybová zložka, ktorá predstavovala pohyb hráča. Posledný experiment bola akumulácia snímok, čo predstavuje modifikáciu prvého experimentu.

Vzhľadom na nedostatočný počet vstupných dát je zložité objektívne vyhodnotiť úspešnosť jednotlivých experimentov. Na druhej strane je možné poukázať na rôzne spôsoby ako pristupovať k problematike. V prípade rozšírenia dátovej sady sa možne stať že úspešnosť experimentov bude odlišná.

V prvom experimente bolo zistené, že neurónová sieť vykazuje minimálne množstvo falošne pozitívnych (*False Positive*) sekvencií. Tento jav je obzvlášť dôležitý pri detekcií rohových kopov. Na druhej strane boli sekvencie, ktoré nie úplne jednoznačne determinovali naozaj pozitívne (*True Positive*) sekvencie, hlavne z dôvodu nízkej variability dát. Na základe toho je možné upraviť vstup do klasifikátoru, ktorý môže byť poskladaný z nižšieho počtu snímok. Následne klasifikátor lepšie detekuje sekvencie s krátkym úsekom.

Druhý experiment s veľkou presnosťou určil naozaj pozitívne sekvencie. Menším nedostatkom tohto experimentu bolo generovanie malého množstva falošne pozitívnych sekvencií – a to hlavne v prípade, ak situácie v oblasti rohového kopu boli podobné detekovanej udalosti. Takto falošne detekované udalosti je možné eliminovať buď posunutím prahu na vyššiu hodnotu, alebo rozšírením dátovej sady.

Tretí experiment mal najmenšiu úspešnosť z pomedzi všetkých experimentov. Toto je možné odôvodniť tým, že obsahoval male množstvo vstupných dát, pretože boli redukované metódou akumulácie snímok. Okrem toho v niektorých sekvenciach nebola dostatočne zachytená pohybová časť, čo zapríčinilo vznik šumu.

Vzhľadom na vyššie uvedené experimenty je možné konštatovať že experiment č. 1 a experiment č 2 vykazovali najlepších výsledkov. Preto v pokračovaní vývoja detekcie udalostí je odporučané isť cestou prvých dvoch experimentov.

Všetky experimenty boli realizované na vzdialenom počítači, ktorý disponoval nasledujúcim hardware:

Tab. 5.6: Dostupný hardware na trénovanie neurónovej siete

Grafická karta (GPU)	Nvidia GeForce GTX 1060 6 GB
Procesor (CPU)	Intel Core i5-7400
Pamäť RAM	16 GB

V tabuľke 5.7 sú uvedené výsledky experimentov. Okrem úspešnosti sú v tabuľke zobrazené aj ďalšie metriky, ako je presnosť precíznosť atď. V každom experimente boli vyhodnotené výsledky naprieč trom datasetom a celkový výsledok bol vypočítaný ako ich priemer.

Tab. 5.7: Porovnanie výsledkov z oboch experimentov

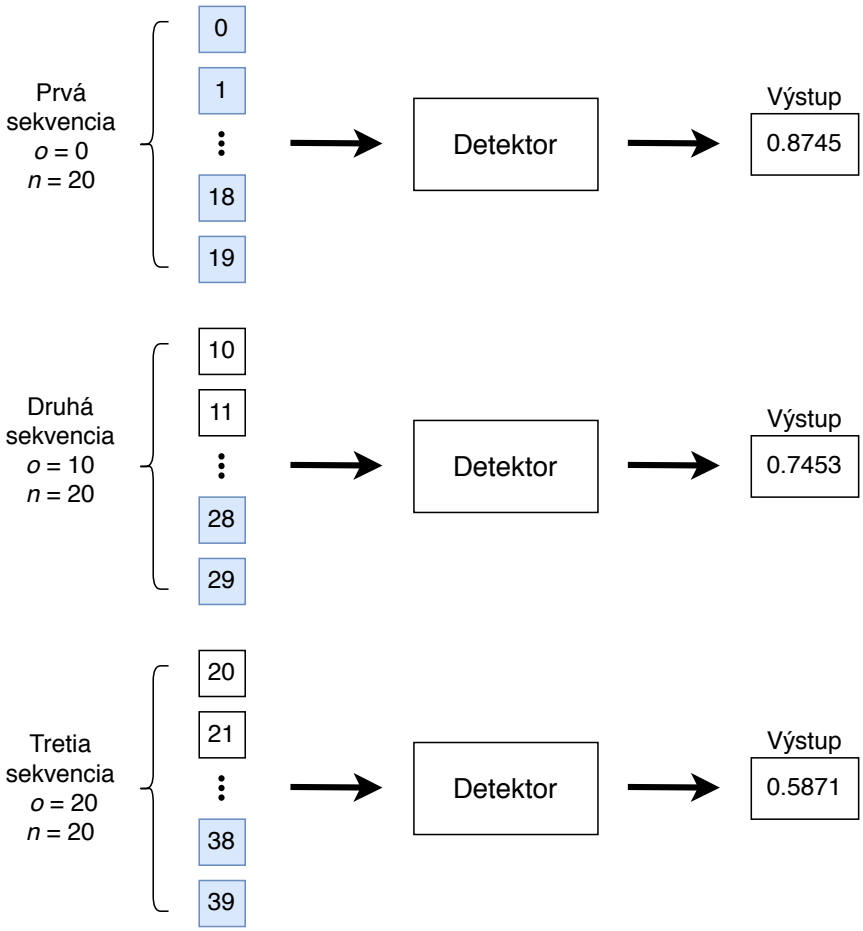
Model	**Dataset**	**Presnosť**	**Precíznosť**	**Úplnosť**	**F1-skóre**
Inception	dataset 1	0.9076	0.9724	0.9057	0.9379
	dataset 2	0.9327	0.9627	0.9241	0.9430
	dataset 3	0.9703	0.9798	0.9483	0.9638
	priemer	**0.9369**	**0.9716**	**0.9260**	**0.9482**
Inception (akum.)	dataset 1	0.8816	0.8973	0.8843	0.8908
	dataset 2	0.9014	0.8977	0.9067	0.9022
	dataset 3	0.8867	0.8967	0.9012	0.8989
	priemer	**0.8909**	**0.8972**	**0.8974**	**0.8973**
C3D	dataset 1	0.9025	0.9438	0.9113	0.9273
	dataset 2	0.9296	0.9519	0.9205	0.9359
	dataset 3	0.9307	0.9682	0.9311	0.9493
	priemer	**0.9209**	**0.9546**	**0.9210**	**0.9375**
Conv3D	dataset 1	0.9513	0.9838	0.9371	0.9599
	dataset 2	0.9025	0.9527	0.9144	0.9332
	dataset 3	0.8961	0.9483	0.9272	0.9376
	priemer	**0.9166**	**0.9616**	**0.9262**	**0.9436**
LSTM	dataset 1	0.9297	0.9783	0.9216	0.9491
	dataset 2	0.8987	0.9517	0.9173	0.9342
	dataset 3	0.8892	0.9328	0.9278	0.9303
	priemer	**0.9059**	**0.9543**	**0.9221**	**0.9379**

6 Detektor udalostí

Poslednou časťou diplomovej práce bolo vytvorenie algoritmu na automatickú detekciu udalostí. Tento nástroj môže mať široké využitie. Napríklad detekcia udalostí zo živého prenosu nejakého futbalového zápasu. Dôležité je, aby detektor spracovával dostatočne rýchlo vstup a rovnako výstup, ktorý je následne prezentovaný používateľovi.

Druhým možným využitím takéhoto detektora je spätná detekcia z videozáznamov, tj. z vyhotoveného videozáznamu sa anotujú jednotlivé udalosti a uložia sa do príslušného formátu. V tomto prípade nie je rýchlosť prioritou.

V oboch prípadoch bola použitá technika **posuvného okna** (*Sliding Window*). Na obrázku 6.1 je znázornený mechanizmus posúvného okna. Veľkosť okna $\boldsymbol{n}$ je voliteľná a jej hodnota závisí od použitej architektúry. Posun začiatku sekvencie $\boldsymbol{o}$ bol vypočítaný ako polovica dĺžky sekvencie: $\mathbf{o = n/2}$.



Obr. 6.1: Praktická ukážka procesu posuvného okna

6.1 Anotácia udalostí z videozáznamov

Jedna z možností využitia tohto algoritmu je realtime detekcia udalostí. Vstupom do detektora je multimediálny obsah, ktorý tvorí živé vysielanie z futbalového podujatia. Výstupom je anotácia udalosti z danej vstupnej sekvencie.

Dáta získané zo snímača (*Raw Data*) umiestneného na hracej ploche, je potrebné predspracovať na vstup do neurónovej siete. Tento postup je podobný procesu predspracovania dát, ktorý bol vysvetlený v kapitole 3 a skladá sa z nasledujúcich krokov:

1. **zostavenie sekvencie snímok** – v prvom kroku sú zozbierané jednotlivé snímky. Vzhľadom k redundancií snímok pri vyššej snímkovej frekvencii a optimalizácii rýchlosti detekcie sú snímky volené s určitým krokom.
2. **orezanie oblasti záujmu** – druhým krokom je orezanie oblasti, v ktorej sa daná udalosť odohráva.
3. **horizontálne preklopenie** – tento krok závisí od pozície danej udalosti. Táto operácia je vykonávaná len v prípade, ak sa udalosť vyskytuje na pozícií opačnej, ako je hlavná.
4. **zmena rozmerov vstupu** – ďalej je možné zmeniť rozmer vstupu, tj. prispôsobiť ho k požadovanému rozmeru vstupu do neurónovej siete.
5. **normalizácia pixelov**.

Na obr. 6.2 je ukážka výstupu z detektora, kde detekovanou udalosťou bol rohový kop. Vložené textové pole informuje používateľa o aktuálnej udalosti. V zátvorke je uvedená hodnota klasifikácie.

Obr. 6.2: Ukážka z výstupu detektora udalostí

Záver

Cieľom diplomovej práce bolo naštudovať metódy detekcie udalostí z video sekvencií zameraných na futbalové zápasy. Na začiatku bolo potrebné zoznámiť sa s postupmi a metódami extrakcie informácií z videa. Vybrané metódy boli následne implementované, pričom získané informácie z video sekvencií boli použité ako vstup pre algoritmus automatickej detekcie udalostí. Z dostupných video sekvencií bola pripravená tréningová a testovacia sada dát. Nakoniec bol navrhnutý algoritmus neurónovej siete, ktorý je schopný automaticky rozpoznávať jednu udalosť – rohový kop. V poslednej časti boli spracované a porovnané výsledky experimentov navrhnutého riešenia.

Prvá kapitola práce bola venovaná úvodu do problematiky detekcie udalostí, zatiaľ čo druhá kapitola sa zamerala na teoretickú analýzu hlbokých neurónových sietí na účel detekcie udalostí.

V praktickej časti bol najprv vytvorený nástroj na spracovanie videozáznamov. Na začiatku bolo potrebné vyrezať oblasť videozáznamu, v ktorej sa detekovaná udalosť odohrávala. Pri rohových kopoch sa táto oblasť nachádza v rohovom štvrťkruhu. Ďalším postupom bola segmentácia takto vytvorených videoklipov na úseky s dĺžkou 2 sekundy, čím vznikne séria krátkych sekvencií. V poslednom kroku boli z každého videoklipu extrahované jednotlivé snímky.

Po úspešnom spracovaní dátovej sady boli implementované zvolené architektúry neurónovej siete, na ktorých boli realizované dva experimenty. Prvý experiment pracoval so statickou informáciou z jednej snímky. Druhý experiment zachytával pohybu zložku pomocou časopriestorových dát. Do prvého experimentu bol použitý model 2D konvolučnej neurónovej siete Inception, zatiaľ čo v druhom experimente boli implementované dve architektúry 3D konvolučných sieti a jeden model rekurentnej LSTM neurónovej siete.

Funkčnosť experimentov bola overená pomocou vytvoreného nástroja – detektora udalostí, ktorého výstupom boli anotácie rohových kopov. Z výsledkov sa dá usúdiť, že detektor úspešne detekuje udalosť a je možné ho aplikovať do procesu ďalšieho vývoja. Okrem iného, algoritmus pracuje realtime, čo predstavuje možnosť implementácie na živé futbalové prenosy.

V snahe dosiahnuť lepšie výsledky detekcie je potrebné v prvom rade rozšíriť dátovú sadu, prípadne upraviť veľkosť rozmeru vstupu. Práca je spracovaná tak, že je možné jednoducho na ňu nadviazať, napr. pridaním nových udalostí.

Vďaka tejto práci je možné získať nové poznatky o problematike, ktoré môžu byť nápomocné pri ďalšom vývoji systému automatickej detekcie udalostí vo futbalovom zápase.

Literatúra

[1] Trainervoetbal: *Position in defence area* [online]. Poslední aktualizace 23. 5. 2017 [cit. 7. 11. 2019]. Dostupné z URL: <http://www.trainervoetbal.be/index.php/component/jdownloads/category/15-software?Itemid=-1>.

[2] Montoliu, R.; Martín-Félez, R. a Martínez-Usó, A.: *Team activity recognition in Association Football using a Bag-of-Words-based method* [online]. Poslední aktualizace Jún. 2015 [cit. 8. 11. 2019]. Dostupné z URL: <https://www.sciencedirect.com/science/article/pii/S0167945715000494#b0005>.

[3] Yosinski, J.; Clune, J. a Bengio, Y.: *How transferable are features in deep neural networks?* [online]. Dept. Computer Science, 2014 Poslední aktualizace 6. Nov. 2014 [cit. 25. 4. 2020]. Dostupné z URL: <https://arxiv.org/pdf/1411.1792.pdf>.

[4] Szegedy, C., Vanhoucke, V., Ioffe, S.: *Rethinking the Inception Architecture for Computer Vision* [online]. University College London London: 2015, poslední aktualizace 11. 12. 2015 [cit. 17. 11. 2019]. Dostupné z URL: <https://arxiv.org/pdf/1512.00567v3.pdf>.

[5] Tsang, S.: *GoogLeNet (Inception v1)— Winner of ILSVRC 2014 (Image Classification)* [online]. Poslední aktualizace 24. 8. 2018 [cit. 17. 11. 2019]. Dostupné z URL: <https://medium.com/coinmonks/paper-review-of-googlenet-inception-v1-winner-of-ilsvlc-2014-image-classification-c2b3565a64e7>.

[6] Tran, D., Bourdev, L., Fergus, R.: *Learning Spatiotemporal Features with 3D Convolutional Networks* [online]. Dartmouth College Hanover: 2015, [cit. 23. 4. 2020]. Dostupné z URL: <https://arxiv.org/pdf/1412.0767.pdf>.

[7] Burton, W.: *2D or 3D? A Simple Comparison of Convolutional Neural Networks for Automatic Segmentation of Cardiac Imaging* [online]. University of Denver Denver: 2019, poslední aktualizace 21. 4. 2019 [cit. 23. 4. 2020]. Dostupné z URL: <https://towardsdatascience.com/2d-or-3d-a-simple-comparison-of-convolutional-neural-networks-for-automatic-segmentation-of-625308f52aa7>.

[8] Shuiwang, J., Wei X., Ming Y.: *3D Convolutional Neural Networks for Human Action Recognition* [online]. Arizona State University Arizona. [cit. 23. 4. 2020]. Dostupné z URL:
<https://www.dbs.ifi.lmu.de/~yu_k/icml2010_3dcnn.pdf>.

[9] Hochreiter, S. a Schmidhuber J.: *LONG SHORT-TERM MEMORY* [online]. Technická Univerzita Mníchov Mníchov: 1997, [cit. 26. 4. 2020]. Dostupné z URL:
<http://www.bioinf.jku.at/publications/older/2604.pdf>.

[10] Nguyen, M.: *Illustrated Guide to LSTM's and GRU's: A step by step explanation* [online]. Poslední aktualizace 24. 10. 2018 [cit. 26. 4. 2020]. Dostupné z URL:
<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>.

[11] Olah, C.: *Understanding LSTM Networks* [online]. Poslední aktualizace 27. 8. 2015 [cit. 26. 4. 2020]. Dostupné z URL:
<https://colah.github.io/posts/2015-08-Understanding-LSTMs>.

[12] Kalchbrenner, N., Danihelka, I. a Graves, A.: *Grid Long Short-Term Memory* [online]. Poslední aktualizace 6. 7. 2015 [cit. 26. 4. 2020]. Dostupné z URL:
<https://arxiv.org/pdf/1507.01526v1.pdf>.

[13] p1sforsoccertalk: *The penalty kick* [online]. Poslední aktualizace 12. 11. 2015 [cit. 10. 11. 2019]. Dostupné z URL:
<https://p1sforsoccertalk.files.wordpress.com/2015/11/walker-zim-pk-sequence_sm.jpg>.

[14] Ghosh, R.: *Deep Learning for Videos: A 2018 Guide to Action Recognition* [online]. Poslední aktualizace 11. 6. 2018 [cit. 10. 11. 2019]. Dostupné z URL:
<http://blog.qure.ai/notes/deep-learning-for-videos-action-recognition-review>.

[15] Rojas, R.: *Lucas-Kanade in a Nutshell* [online]. Dept. of Computer Science, Freie Universitat Berlin [cit. 10. 11. 2019]. Dostupné z URL:
<http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/tutorials/Lucas-Kanade2.pdf>.

[16] Berthold, K.P., Schnuck, H., Schnuck, B.: *Determining Optical Flow* [online]. Massachusetts Institute of Technology, Cambridge: 2009, [cit. 10. 11. 2019]. Dostupné z URL:

<http://image.diku.dk/imagecanon/material/HornSchunckOptical_Flow.pdf>.

[17] Karpathy, A,; Toderici, G.; Shetty ,S.: *Large-scale Video Classification with Convolutional Neural Networks* [online]. Stanford University California: jún, 2014, [cit. 12. 11. 2019]. Dostupné z URL: <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42455.pdf>.

[18] Simonyan, K.; Zisserman, A.: *Two-Stream Convolutional Networks for Action Recognition in Videos* [online]. University of Oxford. Oxford: 2014, [cit. 13. 11. 2019]. Dostupné z URL: <https://arxiv.org/pdf/1406.2199.pdf>.

[19] Carreira, J.; Zisserman, A.: *Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset* [online]. Department of Engineering Science. Oxford: 2018, [cit. 16. 11. 2019]. Dostupné z URL: <https://arxiv.org/pdf/1705.07750.pdf>.

Zoznam symbolov, veličín a skratiek

Adam	Adaptive Moment Estimation – algoritmus minimalizácie chyby
AUC	Area Under the Curve – hodnota pod ROC krivkou
CNN	Convolutional Neural Network – konvolučné neurónové siete
CV	Computer Vision – počítačové videnie
FPR	False Positive Rate – počet falošne pozitívnych predikcií
FPS	Frames Per Second – počet snímok za sekundu
GAP	Global Average Pooling – vrstva spriemerovania
GPU	Graphics Processing Unit – grafický procesor
LSTM	Long Short-Term Memory – rekurentná neurónová sieť
ReLU	Rectified Linear Unit – aktivačná funkcia
RGB	Red Green Blue – aditívny farebný model
RMSprop	Root Mean Square Propagation – algoritmus minimalizácie chyby
RNN	Recurrent Neural Network – rekurentné neurónové siete
ROC	Receiver operating characteristic – graf ROC krivky
SGD	Stochastic Gradient Descent – algoritmus minimalizácie chyby
TPR	True Positive Rate – počet naozaj pozitívnych predikcií

Zoznam príloh

A Návod na spustenie aplikácie **62**
A.1 Predspracovanie dát 62
A.2 Neurónová sieť 63
A.3 Detektor 63

B Obsah priloženého CD **64**

A Návod na spustenie aplikácie

V tejto prílohe je návod na spustenie celej praktickej časti implementovanej v rámci diplomovej práce. Návod sa skladá z troch častí:

1. Predspracovanie vstupných dát
2. Spustenie neurónovej siete a vyhodnotenie výsledkov
3. Použitie detektora udalostí

Aplikáciu je možné spustiť pomocou súboru Makefile, v ktorom sú definované príkazy, podľa ktorých sa dá spustiť jednotlivá časť práce.

A.1 Predspracovanie dát

V prvom kroku je potrebné inicializovať priečinky potrebné na spustenie aplikácie nasledujúcim príkazom:

```
$ make init
```

Orezanie oblasti záujmu a zníženie snímkovej frekvencie je možné realizovať pomocou príkazu:

```
$ make crop ARGS="--fps 25 -p bottom --b_size 150"
```

Výstupom predchádzajúceho kroku sú predspracované videoklipy, ktoré je potrebné ďalej segmentovať na úseky s pevne definovanou dĺžkou. Parameter *–**step*** určuje krok v rámci rozsahu celkovej dĺžky videozáznamu.

```
$ make segment ARGS="-s 2 -p bottom --step 2"
```

Ďalším krokom je extrahovanie snímok z vytvorenej video sekvencie. Tento proces je možné realizovať buď individuálnou extrakciou po sebe idúcich snímok, alebo pomocou akumulácie snímok. Parameter ***-f*** zapína použitie horizontálneho preklopenia snímok.

```
$ make extract ARGS="-f"
```

Z extrahovaných snímok budú následne vytvorené dve množiny: trénovacia a testovacia. Rozdelenie snímok do príslušných sád je uskutočnené pomocou prepínača *–**split***. Presunutie rozdelených snímok do dvoch separátnych zložiek je vykonané prepínačom *–**move***. Parameter ***-l*** definuje použitie identifikátorov štadiónov, ktoré budu použité do testovacej sady.

```
$ make data_organizer ARGS="--split --move -l 7 9 10 28""
```

A.2 Neurónová sieť

Po vytvorení trénovacej a testovacej sady je možné spustiť trénovanie neurónovej siete. V rámci praktickej časti boli implementované viaceré architektúry.

Spustenie prvého experimentu (model Inception) je možné realizovať nasledujúcim príkazom: parameter ***-b*** definuje batch, –***shape*** je rozmer vstupu do neurónovej siete. Ďalej ***-e*** znamená počet krokov v rámci etapy a ***-v*** je počet validačných krokov. Nakoniec je pomocou ***-o*** je možné zvoliť optimalizátor a ***-d*** zapína vrstvy Dropout.

```
$ make train_cnn ARGS="-b 64 --shape 150 -e 100 -v 10 -o adam -d"
```

Ďalší príklad predstavuje spustenie druhého experimentu, ktorý zahŕňa 3D konvolučné neurónové siete. Parameter ***-m*** predstavuje model architektúry. Možnosti, ktoré môže tento parameter nadobúdať, sú: **lstm**, **c3d** a **conv_3d**. Prepínač ***-s*** špecifikuje dĺžku sekvencie.

```
$ make train ARGS="-b 32 -m c3d -s 15 -o adam"
```

Pred samotným spustením rekurentnej neurónovej siete LSTM je potrebné extrahovať príznaky za pomoci modelu Inception. Extrakcia týchto príznakov sa spúšťa nasledujúcim príkazom:

```
$ make feature_extractor ARGS="-s 25 --shape 150"
```

Vyhodnotenie výsledkov neurónovej siete možno spustiť pomocou príkazu:

```
$ make evaluate ARGS="-w dataset1_inception.033-0.12.hdf5"
```

Pre detailnejšie spracovanie výsledkov modelu Inception je možné zavolať príkaz:

```
$ make validate_cnn ARGS="-w dataset2_inception.033-0.12.hdf5"
```

Podobným príkazom je možné zvalidovať aj architektúru z druhého experimentu:

```
$ make validate_rnn ARGS="-m c3d -w c3d.005-0.044.hdf5 -s 25"
```

A.3 Detektor

Nakoniec je možné interpretovať výsledky pomocou nastroja vytvoreného n automatickú detekciu udalostí. Nasledujúci príkaz môže pracovať realtime:

```
$ make detect ARGS="-f file --window 10 --fps 12 -w inception
  .043-0.06.hdf5"
```

Tento príkaz pracuje s existujúcim videozáznamom:

```
$ make annotate ARGS="--window 25 -w lstm.0.077.hdf5 --fps 12 -m
  lstm --batch 32"
```

B Obsah priloženého CD

Na CD sa nachádza aplikácia neurónovej siete vytvorená v programovacom jazyku Python. Samotný projekt je testovaný v operačnom systéme Linux. Súčasťou CD je aj elektronická verzia práce.

- `/` ..... koreňový adresár priloženého CD
  - `1. diplomka` ..... hlavný priečinok s projektom
    - `data` ..... priečinok obsahujúci vstupné dáta
    - `detector` ..... detektor udalostí
      - `capture.py` ..... použitie vstupného záznamu
      - `event_detector` ..... implementácia detektora
      - `main.py` ..... spustenie detektora
      - `models.py` ..... interface modelov
    - `network` ..... neurónová sieť
      - `augmentator`
      - `feature_extractor` ..... extrakcia príznakov
        - `extract_features.py`
        - `extractor.py`
      - `data.py` ..... načítanie datasetu
      - `data_organizer.py` ..... vytvorenie trénovacej a testovacej sady
      - `evaluate.py` ..... vyhodnotenie úspešnosti klasifikátora
      - `models.py` ..... použité architektúry
      - `train.py` ..... trénovanie neurónovej siete
      - `train_cnn.py` ..... trénovanie neurónovej siete
      - `validate_cnn.py` ..... validácia výsledkov
      - `validate_rnn.py` ..... validácia výsledkov
    - `preprocessing`
      - `coordinates`
      - `corners` ..... označenia anotácií rohového kopu
      - `motion`
        - `motion_estimation.py` ..... pomocné funkcie na analýzu pohybu
      - `crop` ..... orezanie oblasti záujmu
      - `extractor.py` ..... extrakcia snímkov z video sekvencie
      - `filter.py` ..... filtrovanie nežiaducich dát
      - `segment.py` ..... segmentácia videozáznamu
    - `.dockerignore`
    - `.gitignore`
    - `constants.py`
    - `Dockerfile`
    - `Dockerfile.gpu`
    - `Makefile` ..... príkaz na spustenie programu
    - `README.md`
    - `requirements.txt`
  - `2. Elektronická verzia` ..... elektronická verzia diplomovej práce
    - `diplomová-práca.pdf`